Dynamic Aspects of Design Cognition:

Willemien Visser

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Dynamic Aspects of Design Cognition: Elements for a Cognitive Model of Design

Willemien Visser

N° 5144

Mars 2004

THÈME 3A

ISSN 0249-6399

Dynamic Aspects of Design Cognition: Elements for a Cognitive Model of Design

Willemien Visser[1]

Thème 3A – Databases, Knowledge Bases, Cognitive Systems
Projet EIFFEL

Rapport de recherche n° 5144 – Mars 2004 - 116 pages

Abstract: This text adopts a cognitive viewpoint on design, focusing on individually conducted activities actually implemented in professional, industrial design projects. It presents elements for a cognitive descriptive model of design that, on the one hand, furthers our understanding of design, and on the other hand, offers elements to people who wish to use such knowledge in order to advance education and practice of professional designers. The text is especially concerned with dynamic aspects of design —that is, it focuses on the activity implemented by designers, especially the cognitive processes and/or strategies they use— rather than with static aspects. Section 1 presents the classical cognitive viewpoint on design, that is, the symbolic information-processing (SIP) approach, represented by Herbert A. Simon. Section 2 focuses on the main alternative to the SIP approach for design, i.e. the "situativity" (SIT) approach, mainly represented by Donald Schön. Section 3 is the main division of this text. It presents nuances and critiques with respect to both SIP and SIT approaches, and completes and integrates these two approaches into our own cognitively oriented dynamic approach to design.

Keywords: Cognitive design studies, Design activity, Design models, Individual design, Cognitive psychology, Problem solving, Symbolic information-processing, Situativity, Herbert A. Simon

[1] EIFFEL "Cognition & Cooperation in Design" - Rocquencourt B.P.105 - 78153 Le Chesnay Cedex (France) – Willemien.Visser@inria.fr

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau – Rocquencourt – B.P. 105 – 78153 Le Chesnay Cedex (France)
Téléphone : +33 (1) 39 63 55 11 – Télécopie : +33 (1) 39 63 58 92

Aspects dynamiques de la cognition en conception : Éléments pour un modèle cognitif de la conception

Résumé : Ce texte adopte un point de vue cognitif sur la conception, en se centrant sur des activités individuelles conduites effectivement dans des projets de conception industriels et professionnels. Il présente des éléments pour un modèle descriptif de la conception qui, d'une part, contribue à la compréhension de la conception et, d'autre part, fournit des éléments à ceux qui veulent utiliser ce type de connaissances pour développer la formation et la pratique des concepteurs professionnels. On s'intéresse en particulier aux aspects dynamiques de la conception — c'est-à-dire,à l'activité telle qu'elle est mise en œuvre effectivement par des concepteurs, notamment les processus et stratégies cognitifs qu'ils utilisent— plutôt qu'aux aspects statiques. La Section 1 présente le point de vue cognitif classique sur la conception, c'est-à-dire l'approche du traitement symbolique de l'information (TSI), représentée par Herbert A. Simon La Section 2 introduit l'approche qui constitue la principale alternative à l'approche TSI de la conception, c'est-à-dire l'approche "située" (SIT), représentée principalement par Donald Schön. La Section 3 est la partie centrale du texte. Elle présente des nuances et des critiques au sujet des approches TSI et SIT, et les complète et intègre dans notre approche cognitive dynamique de la conception.

Mots clés : Etudes cognitives de la conception, Activité de conception, Modèles de conception, Conception individuelle, Psychologie cognitive, Résolution de problèmes, Traitement symbolique de l'information, Action et Cognition situées, Herbert A. Simon

Contents

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau – Rocquencourt – B.P. 105 – 78153 Le Chesnay Cedex (France)
Téléphone : +33 (1) 39 63 55 11 – Télécopie : +33 (1) 39 63 58 92

"Only human pride argues that the apparent intricacies of our path stem from a quite different source than the intricacy of the ant's path". (Simon, 1999, p. 82)

1 Introduction

Design is an important, all-pervading domain of human activity. Not only new sport cars are the object of design, but also artefacts as diverse as traffic signals (Bisseret, Figeac-Letang, & Falzon, 1988), meals (Byrne, 1977), route plans (Chalmé, Visser, & Denis, in press; Hayes-Roth & Hayes-Roth, 1979), and, of course, software (Détienne, 2002a). This text examines design from a *cognitive* point of view, i.e., it focuses on the types of cognitive processes and the types of knowledge used by designers. Analysed from this viewpoint, design projects constitute a particular type of *problems* and involve a rather complex *activity*, which are of interest, first, to cognitive psychology and cognitive ergonomics, but also to other disciplines that examine design from other viewpoints, such as design methodology, Artificial Intelligence and engineering design disciplines.

Even if cognitively oriented studies of design have been developed from the nineteen-sixties on, nevertheless, their results still seem to be considered only in rather restricted circles. In the domain of HCI, e.g., the relevance of cognitively oriented design studies is of course recognised by many authors, but most authors don't show great familiarity with the existing tradition and the research that has already been realised in the field of cognitive design studies. Yet, HCI is undeniably a field with close relations to cognitive psychology and cognitive ergonomics.

Our aim in this text is to present elements for a cognitive descriptive model of design that, on the one hand, furthers our understanding of design, and on the other hand, offers elements to people wanting to use such knowledge in order to advance the education and the practice of professional designers. The text is especially concerned with *dynamic* aspects of design —that is, it focuses on the activity implemented by designers, especially the cognitive processes and/or strategies they use— rather than with static aspects. Therefore, instead of describing the content of knowledge used, this text focuses on the *actual use* of this knowledge.

The text focuses on *individually conducted* activities implemented in *professional, industrial design* projects. Much underlying data indeed is based on our *observational studies* conducted in industry.

1.1 Outset of the paper

This text is organised in the following way. The present *Introduction* introduces the main types of design models and some historical pointers, and presents two series of preliminary definitions. These definitions are given in order to reduce sources of misunderstanding between readers from different domains. They cover the domains of "problem solving" and "data collection". *Section 1* describes the classical cognitive viewpoint on design, that is, the symbolic information-processing (SIP) approach, represented by Herbert A. Simon. *Section 2* focuses on the main alternative to the SIP approach for design, i.e. the "situativity" (SIT) approach, mainly represented by Donald Schön. *Section 3* is the main division of this text. It describes the approach to design that we have been developing for some 20 years now[2]. It presents nuances and critiques with respect to both SIP and SIT approaches, and completes and integrates these two approaches into our own cognitively oriented dynamic approach to design. Special attention is

[2] With respect to "we" and "I", the convention adopted in this text is the following. When the author is defending views that she supposes that she shares with most of her colleagues in the domain of cognitive psychology and cognitive ergonomics (especially with her colleagues of the EIFFEL research group, see e.g., Détienne & Falzon, 2001), "we" will be used. "We" will also be used in the classically publication style, i.e. to refer to the author announcing sections to come, or recalling issues already handled. First-person articles ("I", "me"" and "mine") will only be adopted to present opinions that colleagues might not be willing to endorse.

paid to software design, one of the domains in which we have been conducting several empirical studies. In the *Conclusion*, we discuss the different approaches to design presented in the text and we propose some unexplored topics and questions to be examined in further research.

1.2 Some preliminary remarks

Some pivotal points of this text that will be discussed in detail later on, are addressed briefly in this section, in order to discard, if possible, likely misunderstandings.

1.2.1 Selective focus on cognitive aspects of design

This text, of course, doesn't handle with all aspects of viewpoints on design that cognitively oriented —abbreviated into "cognitive" in the rest of this text.
With respect to the objects of design, that is, artefacts, we focus on product design; this text is not going to examine the design of production systems (for design of production systems conceived from an "ergonomic intervention" viewpoint, see Daniellou, 1999; Desnoyers & Daniellou, 1989).
We do not particularly address design education or design assistance.
Focusing on cognitive aspects of design, this text does not deal with emotional or socio-emotional factors (see Carroll, 2000).

1.2.2 Design as a type of problem solving activity

A position that is central in this text is the analysis of design as a problem solving activity. Misunderstandings evoked by this approach are due in part to the particular, technical sense of "problem" that is adopted in cognitive psychology. It will be presented and discussed in the *Preliminary definitions* section. Anticipating this upcoming presentation and discussion of our position, we may state that "design as problem solving" especial refers to the following opposition. As problem solving, design amounts to elaboration of procedures rather than to their execution.
Of course, design is not "only" problem solving. As noticed above already, design may also involve emotional and socio-emotional factors —probably influencing cognitive aspects. Especially in collective design situations, design also involves interaction with other people, particularly colleagues and clients. As this text focuses on individually conducted design, these aspects are less central.

1.2.3 Elements for a cognitive model of design

This paper presents elements for a cognitive model of design, which is descriptive rather than prescriptive —and not yet predictive. It does not pretend to completely specify such a model. Some components are being more detailed than other ones. In our own studies, the organisational and reuse components have been especially analysed and described in detail. Our work has led to qualify design as an opportunistically organised activity, in short, an "opportunistic" activity in which reuse plays an particularly important role. Other components require that the already available data be completed and formalised, and still other parts even require that research be initiated.
Before presenting our own approach, this paper discusses the two main "families" of models in the domain (the two dominant paradigms), i.e. the symbolic information processing (SIP) and the situativity (SIT) paradigms (Dorst, 1997). For ill-defined problems such as design, information processing is, in our view, not entirely adequate. Alternative proposals, especially those focusing on situativity, however, do not yet seem appropriate either. This twofold judgement will be discussed below. Our approach will be to complete and integrate the two types of models.
We selected the blackboard approach for modelling design[3]. We adopted it in the past for representing design organisation (see also Bisseret et al., 1988; Hayes-Roth & Hayes-Roth, 1979;

[3] This text does not provide a discussion of the differences between models, frameworks, and theories. This paragraph only presents some keywords concerning our position to this respect.

Visser, 1988c, 1990, 1994a; Whitefield, 1986). The blackboard model of problem solving is indeed a special case of opportunistic problem solving (Nii, 1986a, 1986b) and, as noticed already, opportunism is one of the central features of design, characterising both its organisation, and designers' reasoning style and knowledge use.
We are of course conscious of the tension and the —difficult to attain— equilibrium between two types of risks to diminish the ecological validity of the ultimate model as they are related to two types of data-collection situations:

- experimental situations that may not be representative of "real" work situations (the ones we are heading); and
- work situations (the "field") whose results may not be generally valid.

Data presented will come mostly from field studies of design activities, but many characterisations of design referred to in the literature do not have such an empirical basis (e.g., Simon's SIP model).
In the engineering domain, researchers often qualify their empirical studies as research on "design thinking" (Buchanan, Spring 1992; Cross, Dorst, & Roozenburg, 1992; Dörner, 1999; Galle, 1996; Rowe, 1987), sometimes distinguishing two kinds of design thinking, i.e. "design reasoning" and "design intuition" (Akin, 1992, p. 39).

1.2.4 Focus on the "early stages" of design

Our focus is largely on what are prescriptive models consider the "early stages" of design, that is, the "preliminary", "logical", or "conceptual" stages in these models that decompose design in several "stages" (see below). Our definition of design from a cognitive viewpoint indeed attributes a central place to the definitional and decisional, creative, conceptual aspects of the activity.

Economic relevance of this focus. Economic analyses of the design process show the importance of research devoted to these early stages. Such analyses have shown that the study of alternatives is then much less expensive (Perrin, 1997 quoted in Perrin, 1999). Several authors advance percentages concerning the corresponding cost and possible gain. They affirm, e.g., that the first 5% of the design process commits more than 75% overall costs in order to market the product (Carrubba, 1993, referred to in Hsu & Liu, 2000; Sharpe, 1995).
In order to be able to examine different possible solutions, to confront the viewpoints of the different stakeholders in the design project, and to attain pondered compromises, these stages should occupy more time. The corresponding supplement in duration and in cost would be largely compensated afterwards. In principle, this is one of the justifications of concurrent engineering (Perrin, 1999).

1.2.5 Focus on the activity

In a cognitive psychology context, our focus on the activity of design refers to several oppositions, especially that between "task" and "activity", which will be discussed in the *Preliminary definitions* section.
In a design-engineering context, focus on the activity pertains to other references. As noticed by Blessing (1994, p. iii), in order to improve the design process, various means have been investigated:

- prescriptive models of the design process;
- handbooks and catalogues containing design knowledge;
- methods to support specific design activities;
- computer-based tools.

Blessing explains why "these means have not had the expected impact on the effectiveness and the efficiency of the design process". She argues that "focus on the design activity rather than on its deliverables… would be the most appropriate approach to improve design" —a position we agree with completely. In addition, Blessing adds, "A focus on the design process can increase

the impact of the existing means". (1994, p. iii) As a conclusion of her Ph.D. thesis work, Blessing (1994) presents a process-based, computerized, design support system.
In a design-engineering vs. cognitive-psychology context, our focus on the design activity refers to the opposition between goal-defined design stages in a prescriptive design process model and the activity that is actually implemented in order to attain the corresponding goals.
In the rest of this text, unless specified otherwise, the "design" referred to (through expressions such as "studies of design", or "design is opportunistic") is the *actual design activity* implemented by designers during their work on design projects. "Actual" design refers to the design activity as "really" implemented by designers in design projects. This *actual design activity* is thus opposed to the design task or the global design process in normatively based descriptions or prescriptions of design methods or design processes.

1.2.6 Focus on individual design

The present text is restricted to individual design. This might be considered an artificial, or even irrelevant approach. Design always involves several people —at least two, a client, and a designer. In spite of this fact, data concerning individual design continue to be particularly relevant, and well on at least three grounds.
First, even if many design projects are undertaken by large teams involving big numbers of designers, engineers and other people; even if discussion, negotiation, cognitive and operative synchronisation play a crucial role in the elaboration and selection of solutions, an important proportion of design activity remains the work of single individuals, especially during distributed design[4] (for software engineering, see e.g. Kitchenham & Carn, 1990). In their "Resume of 12 years interdisciplinary empirical studies of engineering design in Germany" (Pahl, Badke-Schaub, & Frankenberger, 1999) advance[5] that "in design processes individual work dominates to an extent of > 70%, compared to < 30% of teamwork" (p. 490).
In addition, even during co-design[6], cognitive activities in collective design are those implemented in individual design augmented by other activities that are specific to co-operative work (especially co-ordination, communication, management of viewpoints, synchronisation and conflict resolution, largely through argumentative activities). We do not see any evidence to suppose that co-operation modifies the nature of the elementary problem solving processes implemented in design (i.e., solution development and evaluation processes)(Visser, 1993a).
Finally, the development of appropriate work environments, such as shared and private work spaces in (computer-) mediated design, requires the analysis of the articulation between the different forms of reasoning implemented in both individually and collectively conducted activities.

1.2.7 Generic design

Even if often not stated explicitly, the tendency in cognitive design research is to aim generic design models that generalise across design tasks in different domains (Goel, 1994; Thomas & Carroll, 1979/1984). This text adopts also this viewpoint. The underlying belief of this approach is that there are both important similarities between design tasks in different domains and important differences between design tasks and non-design tasks.

---

[4] In distributed design situations, several designers are involved simultaneously, but not together, in the same collective process. They carry out well-determined tasks that have been allocated beforehand to them, and they pursue goals (or at least sub-goals) that are specific to them. One of their objectives is to participate as efficiently as possible in the collective resolution of the problem.
[5] On the basis of the protocols collected during the authors' observational studies in industry (28 weeks) (Personal communication).
[6] In co-design situations, design partners develop the solution together: they share an identical goal and contribute to reach it through their specific competencies. They do so with very strong constraints of direct co-operation in order to guarantee the success of the problem resolution.

We nevertheless suppose that the nature of the design object (artefact) influences the design activity. This means that, in spite of generic design characteristics, there are different "types" or "forms" of design. As far as we know, this assumption has never been the object of specific cognitive analysis. In the *Conclusion,* we will discuss some candidates for dimensions differentiating different "types" of design.

1.3 Models of design

In 1984, that is, 20 years ago, Cross edited a series of papers presenting a history of the *Developments in design methodology*, collecting texts written by the authors who had been involved in the underlying "movement" (Cross, 1984). The collection covers the period that goes from 1962 (when the first Conference on Design Methods marks the birth of design methodology) until 1982 (when the Design Policy Conference signifies the coming of age of design methodology). The organisation of Cross' book reflects the progression of the movement through four stages:

- prescription of an ideal design process (1962-67: the period of "systematic design" proposed by the proponents of the "design methods movement");
- description of the intrinsic nature of design activity (1966-73: design problems were discovered to be not so amenable to systematisation; authors tried to understand their apparent complexity, attributing such in big part to design problems' "ill-structuredness");
- observation of the reality of design activity (late 1970s: methodical collection of data on the actual design activity);
- reflection on the fundamental concepts of design (1972-82: emergence of a more fundamental and philosophical approach to design method).

1.3.1 Prescriptive and descriptive models

Given these developments shown by Cross, one might thus expect that, today, more than thirty years after several of Cross' historical design studies, we are in a new stage, "beyond" "simple" descriptions and observations. One might expect that predictive models have been built, compared, and that a consensus has been attained concerning "the nature" of design. This is, however, not the case at all!
Most cognitively oriented authors distinguish "prescriptive" (or "normative") and "descriptive" models —and continue to collect elements for such descriptive models of design —whereas predictive models are not even yet envisioned in this domain.

Prescriptive models. Prescriptive models of design are developed in order to monitor the design process. Generally linear and sequential, they stipulate that problem solving follows an abstract-concrete axis (going from conceptual to physical specifications) by an iteration of two complementary stages: generation of a solution (that may be partial or intermediate) and its evaluation, which leads to the generation of a better solution, which itself is evaluated, and so on until obtaining the ultimate solution (see Darses, 1997, for a critique of this view).
For engineering design, prescriptive models of design and corresponding methods are exemplified by Pahl and Beitz (Pahl & Beitz, 1977, 1984), especially, but not only, in Germany. Design of software has been prescribed by several models: the classic software life cycle (or "waterfall chart") model (Boehm, 1976), stepwise refinement models (Wirth, 1971)[7], and incremental release models (Basili & Turner, 1975) (see also the IEEE Standard 1074-1997 for Developing Software Life Cycle Processes, and the ISO/IEC International Standard 12207-1995: Information Technology. Software Life Cycle Processes) (Scacchi, 2001).

[7] See our section "Planning and organising design" for Wirth's pragmatic ideas about the actual application of the methods based on the model he proposed.

To prescriptive models and methods, correspond norms, which differ according to countries and their cultures, working and otherwise (norm BS 7000 in Great Britain[8], norm AFNOR X50-127 in France, norm DIN or VDI[9] 2221/2 in Germany)(VDI, August 1987).

Descriptive models. Few global descriptive models of *actual* design have been developed. Proposals for design models generally focus on particular aspects of the design process. Darses (1994) has formulated elements for a model of design based on constraint management, focusing on solution generation. Bonnardel (1992) has analysed the evaluation component of design, which also relies on constraints and/or criteria. Burkhardt and Détienne (Burkhardt, 1997; Burkhardt & Détienne, 1995; Détienne & Burkhardt, 2001) and Visser (Visser, 1999b, 1993c; Visser & Trousse, 1993) have contributed to possible models of reuse in design. We have also proposed elements for the organisational component of a model of design (Visser, 1988c, 1990, 1994a).
In order to formulate global models of actual design, articulating its different components, one needs data with respect to the actual activities carried out by designers and their articulation. Research conducted in cognitive psychology and cognitive ergonomics may provide such data, generally through empirical observational studies, in the field or in controlled experimental conditions in an ecological setting. Such an approach to design is central to this paper. With respect to Cross (1984)' decomposition presented above, research relevant in this context is that corresponding to his stages 2, 3 and 4.

1.3.2 Stage models and process models

Most prescriptive models present design as a series of steps to be followed in a top-down, stepwise manner (Blessing, 1994; Warren & Whitefield, 1987). Indeed, prescriptive models are generally "stage models". Likewise, most stage models are prescriptive, but there are also descriptive, cognitive models that present design as proceeding in a series of consecutive steps. According to Simon's model of design (Simon, 1973/1984, 1999), ill-structured problem solving proceeds through two consecutive stages, first structuring the ill-structured problem, then solving the resulting well-structured problems. This view will be discussed in more detail below.
NOTE: When used in a technical sense, we are going to use the term "stage" for the prescribed steps, and "phase" for periods in the activity that can be observed to actually constitute a separate entity. First, this means that certain stages can correspond to phases (and v.v.).
Second, descriptive models often also present the activity as proceeding through several "phases" or "stages", with the analytical aim of globally rendering the activity, by definition a modelling objective. The view of design as organised —in the sense of *to be* organised— in several consecutive stages should, however, not be confused with a distinction frequently encountered in the cognitive-psychology literature. This is the analysis of problem solving into three "stages" or "phases", i.e. "construction of a problem representation", "solution development" and "solution evaluation" (see e.g. Reimann & Chi, 1989; Richard, 1990; Visser, 1991b). Such decompositions that are useful on an analytical level do not render the actual problem solving activity, which is rarely structured into separate stages, be it two, three or four (see below).
Thomas (1989) proposes a long list of "stages": "problem formulation", "idea generation", "idea evaluation", "solution evaluation", "idea integration", "solution acceptance, or modification". He even decomposes certain stages: problem formulation, e.g., requires "goal setting" and "goal elaboration" (Thomas, 1989). The author notices that this model is rather a normative than a descriptive one. "In particular", he states, "problem finding, problem formulation, and looking

[8] "The management of product design", published by the British Standards Institution in 1989 (according to Pahl & Beitz, 1996, p. xx).
[9] VDI = Verein Deutscher Ingenieure, i.e. Organisation of German Engineers.

back later at the problem solving process are extremely useful but seldom utilized". (1989, p. 325)
Most process models are descriptive, even if many authors developing descriptive models do not present them as process models. Most cognitive models may be qualified as "process models", but their authors generally do not use this terminology.

Stage models. These models describe the design process as a series of stages through which a designer passes. The stages are to be traversed consecutively, and in a particular order. Each stage must be finished before the next one is engaged, and once a stage is finished, one cannot return to it.
A design stage has been defined as a "subdivision of the design process based on the state of the product under development" (Blessing, 1994, p. 10). Design stages generally are characterised in terms of their input and output (or goal). Stage models usually do not describe the activity that is implemented in each stage and that leads from its input to its output. In each stage, a variety of techniques appropriate to the main task are proposed. Stages may be decomposed into finer stages, which may correspond to what may seem "activities" (and which Blessing, 1994 qualifies as such, see, e.g. p. 10 and p. 20). The finer stages may be articulated in an iterative manner, i.e. in cycles.
Several stage models do not merely cover the design process, but the artefact's global lifecycle, from the "policy stage", passing through the macro-stage of "design", to the use or disposal of the product (for a list of stage models and their characteristics, see Blessing, 1994, p. 20).
In the engineering design domain, many models in terms of stages have been proposed. A famous example and reference in the mechanical engineering design domain, is the prescriptive four-stage model by Pahl and Beitz (1977; Pahl & Beitz, 1984)[10]:
- product planning and clarifying the task (elsewhere often called "analysis");
- conceptual design;
- embodiment design (elsewhere called "layout design", "main design", "scheme design", or "draft design");
- detail design.

The output of the second phase, conceptual design, is a design concept that constitutes the input to the third phase.
The third phase has received many other appellations. French (1971) called it "embodiment of schemes". The output of this third phase is a technical description, often in the form of a scale drawing, which, depending on the particular company involved, may be called a "general arrangement", a "layout", a "scheme", a "draft", or a "configuration drawing".
Other authors add as fifth and sixth stages:
- evaluation and decision taking;
- presentation of results.

Another reference in the engineering design literature is the ASE paradigm, which distinguishes three stages, i.e. analysis, synthesis, evaluation. The three-stage ASE framework constitutes the basis of the "systematic design" method and has been widely accepted in the design methodology community. The reference author for the analysis - synthesis distinction in design is John Christopher Jones (see also Asimov, 1962; 1984, Originally published in 1963). Jones was one of the first authors to propose a way of organising the design process so that logical analysis and creative thought —assumed by Jones to be both necessary in design— would proceed in their own different way. By providing systematic methods for keeping data, information, requirements, and so on, outside one's memory, Jones' systematic-design method attempts to leave the designer's mind as free as possible for random, creative ideas or insights.
NOTE: We will see that the distinction between "analysis" and "synthesis" also refers to a problem typology.

[10] This model is similar to the one proposed in 1971 by French (1971).

In the French norm AFNOR X50-127 "Recommendations for obtaining and assuring quality in design", the French norm-editing organism distinguishes three stages: feasibility study, pre-project and project development (see Perrin, 1999, p. 22).
Stage models are also qualified as "linear", as opposed to "non-linear" models. "Though there are many variations of the linear model… its proponents held that the design process is divided into two distinct phases: problem definition and problem solution. Problem definition is an *analytic* sequence in which the designer determines all the elements of the problem and specifies all the requirements that a successful design solution must have. Problem solution is a *synthetic* sequence in which the various requirements are combined and balanced against each other, yielding a final plan to be carried into production". (Buchanan, 1990, p. 355) As an example of a non-linear model, Buchanan presents the "wicked problem" theory developed by Horst Rittel in the middle 1960s, exactly as an alternative to the linear, step-by-step design models of that era (see Cross, 1984).

Process models. These models present activities or actions[11] carried out by a designer in order to realise a task. As noticed above, most cognitive models are such "process models", even if their authors generally do not qualify them as such. The proposition of process models presenting lists of activities or actions is often the work of researchers from engineering design domains.
As a result of her comparative analysis of prescriptive models, Blessing (1994) proposes a "process model" as the most appropriate basis for a support system. It combines explicitly stages and "activities" (see our remark above concerning the "goal-type" character of these activities). It does so for each element of the "product", i.e. for each sub-solution. In a proposal that may be analysed as a combination of Newell and Simon (1972)'s two stages and Pahl and Beitz (1977; 1984)'s four stages, Blessing distinguishes three main stages:
"a problem definition stage resulting in a problem definition and a set of requirements;
a conceptual design stage resulting in a concept (or solution principle);
a detail design stage resulting in a full product description".
"Often [the detail design stage] is divided into embodiment design, in which the concept is developed resulting in the final layout, and detail design in which every component is fixed in shape and form, resulting in a full product description". (p. 10)

1.3.3 Confronting these different models

Data from different sources concur in testifying that prescriptive models do not have —any longer— the power that one may wish to attribute to them, or to their corresponding methods.
Surveys of companies all over the world (Europe, USA, Japan, Korea) have shown that few enterprises have managed to control and improve their design process at the same degree of efficiency as they did for their production process (for references, see Culverhouse, 1995). Contrary to other phases of product development, especially production, there are no norms or company practices for controlling the duration of the design process.
As result of a comparative analysis of the principal prescriptive models, Blessing (1994) asserts that these normative methods are seldom really applied. She concludes that these models provide characteristics of ideal, rather than existing and/or observed design processes.
In a series of empirical studies conducted both in the field and in the research laboratory, a group of German researchers also showed that designers using these methods didn't produce

[11] We use the term "process" (in the expression "design process" or without any other qualification) with the acception that is has in engineering and design-methodology communities, i.e. as referring to all action globally taking place during a design project. Qualified by the attribute "cognitive", it refers to psychological elementary processes (also called "operators", or "mechanisms"), such as memory activation or object identification. We use the term "activity" for a collection of cognitive elementary processes spread out in time, but endowed with a unit and an organisation (see Le Ny, 1991, in Bloch et al., 1991). The cognitive units in which we are particularly interested here, are these activities, such as solution generation and evaluation.

better designs than their colleagues who had long professional experience and who didn't use these methods (Pahl, Badke-Schaub et al., 1999).
Many other cognitive studies of designers' actual activities confirm that the sequential organisation of design such as prescribed by various methodologies, runs up against the actual activity of these designers. The cognitive processes that designers implement to solve design problems, do not follow such a sequence. The formal separation introduced in prescriptive stage models between problem analysis (upstream) and decision-making and action (downstream) is in contradiction with the opportunistic character of the activity. This opportunism of design activity expresses itself, for example, in the intermingling of actual solution generation and solution evaluation activities (Visser, 1994a). The rest of this text will abundantly present this type of results.
With respect to the ASE paradigm, several types of critique have been formulated. In 1998, Gero (, p. 49) states that "the processes involved in this view of designing use a terminology which is no longer widely accepted. The term 'analysis' has been replaced by 'formulation' or similar terms and 'synthesis' is now used to refer to a precursor of evaluation".
A more fundamental critique, already formulated in the early days of design research, is that the analysis-synthesis-evaluation cycle is inappropriate. Designers, rather then traversing these three stages, constantly generate new task goals and redefine task constraints. "Hence 'analysis' is a part of virtually all phases of design. Similarly, 'synthesis' or solution development occurs as early as in the first page of the protocol". (Akin, 1979/1984, p. 205) Normally designers start with a broad, top-down approach to the task. However, "all solutions do not arise from an analysis of all relevant aspects of the problem. Often a few cues in the environment are sufficient to evoke a pre-compiled solution in the mind of the designer. Actually, this is more the norm than a rational process of assembly of parts, as suggested by the term 'synthesis'". (id., p. 206) For other early critical reviews of prescriptive models, see (Carroll & Rosson, 1985; Cross, 1984; Dasgupta, 1989, pp. 31-33; Visser & Hoc, 1990).
Prescriptive models are, nevertheless, still very powerful, both in the form of prescriptive methods and as representations of design that are considered "factual" —even if certain events might be taken as translating the putting into perspective of their power. Such an event was the *Human Behaviour in Design* Symposium (Lindemann, 2003), organised in Pahl and Beitz' Germany with its particularly strong methodological tradition (based on prescriptive stage models), attended, not only by cognitively oriented researchers, but also by design methodologists.
NOTE: If one would wish to analyse design —and other problem solving activities— as taking place through "phases", they would be so numerous that they would be more appropriately considered as very small "steps" during each of which a designer implements a particular activity. Examples of fundamental design activities that occur all through the design process, are activities related to constructing, elaborating, and modifying representations of the problem, that is, representations of the current state of the design, of the requirements and of the goals to reach.

1.4 Some historical pointers

Until some ten to fifteen years ago, most design research was concerned with individual design. In more recent years, an important shift in research on the cognitive aspects of design has consisted in studying, perhaps even to a greater degree, collectively conducted design activities (Béguin, 1994; Blessing, Brassac, Darses, & Visser, 2000; Cross, Christiaans, & Dorst, 1997; Cross, Christiaans, & Dorst, 1996; Détienne, 2002b; Lindemann, 2003) —even if there are also some older studies on cooperation in design projects (see e.g. Klein & Lu, 1989; Visser, 1993a; Walz, Elam, Krasner, & Curtis, 1987)[12]. One may also consult papers presented at conferences

[12] Some rare studies establish a relationship between individual and collective design (Visser, 1993a, 1999a). The 1994 Delft workshop (Cross et al., 1997; Cross et al., 1996) offfered the occasion to compare how an individual and a team solved the same design problem. The organisers indeed provided a group of

such as ECCE —European Conference on Cognitive Ergonomics—and, especially, COOP, the International Conference on the Design of Cooperative Systems, that take place every year since 1994. The papers at the CSCW conferences —which also might seem relevant in this context— are focusing more on technological solutions than on analyses of collective activities —or, if they present analyses of collective work, they do not analyse the underlying cognitive activities.
The change in focus from individual to collective —which has also been observed in other task domains— has been accompanied by the acceptance of design as a central field of research in cognitive ergonomics for which specific analysis methods are being developed (Darses, Détienne, Falzon, & Visser, September 2001). Yet, a great number of studies of design are still being conducted in artificially restricted laboratory conditions, in which the design situation is rather different from that in professional design situations (see e.g. most studies in Design Studies, 1997). Nevertheless, compared to other problem solving activities, design has started to be examined rather frequently in actual working situations. After a few older studies (see e.g. Klein & Lu, 1989; Krasner, 1987; Visser, 1993a; Walz et al., 1987), recently design is even being studied in large-scale industrial settings (Darses, 2002; Détienne & Falzon, 2001; Martin, Détienne, & Lavigne, 2001).
In cognitive psychology and cognitive ergonomics, research on design has been initiated only long time after other work on problem solving. Except for some rare isolated and precursor studies in the sixties and seventies of the twentieth century[13] (Akin, 1979/1984; Byrne, 1977; Darke, 1979/1984; Eastman, 1969; Eastman, 1970; Hayes-Roth & Hayes-Roth, 1979; Lawson, 1979/1984; Marples, 1961; Reitman, 1964, 1965; Thomas & Carroll, 1979/1984)[14], it is only since the eighties that empirical work in this domain has started (Adelson, 1984; Akin, 1986; Baykan, 1996; Bucciarelli, 1984; Carroll, Thomas, & Malhotra, 1980; Chatel & Détienne, 1996; Darses, 1994; Détienne, 1991a; Guindon, Krasner, & Curtis, 1987; Hoc, 1987; Lebahar, 1983; Malhotra, Thomas, Carroll, & Miller, 1980; Michard, 1982; Rist, 1990; Rosson & Alpert, 1990; Schön, 1984; Suwa & Tversky, 1997; Ullman, Staufer, & Dietterich, 1987; Visser, 1987).
The first empirical studies of design concerned *architecture* (see also Cross, 1984; Eastman, 1969; Eastman, 1970)[15](see also Akin, 1979/1984; Akin, 1986; De Vries, 1994; Hamel, 1989; Lawson, 1979/1984) and *mechanical and industrial engineering* (Ullman, Dietterich, & Staufer, 1988; Visser, 1990; Whitefield, 1986). Subsequently, *software* design has become an application domain under study in cognitive design studies (D'Astous, Détienne, Visser, & Robillard, in press; D'Astous, Robillard, Détienne, & Visser, 2001; Détienne, 2002a; Jeffries, Turner, Polson, & Atwood, 1981; Visser, 1987). Today, *engineering* design is one of the central domains in which design is being examined, often in collective settings (Ball, Evans, & Dennis, 1994; Frankenberger & Badke-Schaub, 1999; Martin et al., 2001). Other domains in which empirical studies have examined design from a cognitive point of view are *musical composition* (Reitman, 1964, 1965), *computational geometry algorithm* design (Kant, 1985; Kant & Newell, 1984); *traffic-signal setting* (Bisseret et al., 1988); *computer-network* design (Darses, 1990b, 1990c); *data-base* conceptual modelling (Karsenty, 1991); *industrial design* (Archer, 1965/1984; Cross et al., 1997 ; Cross et al., 1996; see also Dorst, 1997; Visser, 1995b); *composite-structure* design (Bonnardel, 1991a; Visser, 1991b, 1993a). Less common problem solving activities that have been analysed as design are certain *letter-writing and naming activities*, or even *communi-*

design researchers with two videos (and their protocols) corresponding to the conceptual design of a mountainbike attachment as executed by an individual designer and by a team of three designers. Most researchers analysed only one of the videos and/or protocols, but some of them proceeded to comparative protocol analysis (Dwarakanath & Blessing, 1996; Günther, Frankenberger, & Auer, 1996).

[13] See Robert (1979) for references to other early empirical design studies.

[14] Eastman has analysed a protocol collected in a laboratory study concerning an architectural problem. Even if the problem was rather simple, his protocol study constitutes a reference in the domains of empirical studies of design, on the one hand, and ill-defined problems, on the other.

[15] Recently, we have started to study collective design in this domain (Traverso & Visser, 2003).

cation in general (Thomas & Carroll, 1979/1984), *knowledge modelling*[16] (Visser, 1993b), *errand planning* (Chalmé, Visser, & Denis, 2000; Chalmé et al., in press; Hayes-Roth & Hayes-Roth, 1979); *text composition* (Bisseret, 1990; Hayes & Flower, 1980); *meal planning* (Byrne, 1977).

1.5 Our empirical design studies at a glance

As we will regularly refer to our studies for the presentation of examples, Table 1 presents our five main studies of individual design at a glance[17].

1.6 Preliminary definitions

In order to reduce sources of misunderstanding between researchers from different domains, we introduce this section dedicated to terminological issues, in the domains of "problem solving" and "data collection". Those familiar with cognitive psychology or ergonomics may skip this section.

1.6.1 Definitions in the domain of "problem solving"

As mentioned above, a central position in this text is the analysis of design as a problem solving activity. We also referred to the misunderstandings that this approach evokes. That is why we present and discuss the corresponding notions in this section.

A "problem" is not a "difficulty" or a "deficiency". In many pre-scientific texts and contexts, but even in research papers and other scientific publications, the term "problem" is used as a synonym of "difficulty" or "deficiency". It is also often associated with the idea that a "problem" is being solved "once and for all" —something strongly suggested, e.g., by the information-processing approach represented by Newell and Simon (1972) (see below). This leads many authors to consider that design is no problem solving, opposing "problem solving" to

Table 1. Our empirical design studies at a glance

[16] In the KADS knowledge elicitation domain (see Breuker et al., 1987), modeling is considered a task that resembles design (as we did in this Visser, 1993b study).

[17] Visser (1991a) presents an overview of our early research on design in different application domains, that is, software and mechanical design.

REFERENCE USED IN THIS TEXT	Software design	Functional specification	Software debugging	Composite-Structure	("Delft study")
MAIN PUBLICATION FIRST PRESENTED IN – ALSO PRESENTED IN -	(Visser, 1987) (Visser, 1985) (Visser, 1988a)	(Visser, 1990, 1994a) (Visser, 1986a) (Visser, 1988a, 1988c, 1989)	(Visser, 1988b) (Visser, 1986b) (Visser, 1988a)	(Visser, 1991b) (Visser & Bonnardel, 1989)	(Visser, 1995b) (Visser, 1994c)
TASK DOMAIN	Software design: Control program for an automatic machine tool (AMT)	Mechanical design: Functional specification of an automatic machine tool (AMT)	Software design: Control program for an automatic machine tool (AMT)	Composite-structure design (aerospace): Unfurling antenna for a satellite	Industrial design: mountain bike attachment
ROUTINE-NONROUTINE	rather routine	rather routine	rather routine	rather non-routine	rather non-routine
DESIGNER	software engineer (SE)	mechanical design engineer (MDE)	software engineer (SE)	composite-structure designer (CSD)	mechanical design engineer (MDE)
WORKING ALONE / IN TEAM	working alone	foreman of a team	working alone	foreman of a team	working alone
DATA COLLECTION: FIELD/EXPERIM.	field study	field study	field study	field study	experimental study
DATA COLLECTION	Observation and Simultaneous verbalisation: Note taking and Document collection (selective)	Observation and Simultaneous verbalisation: Note taking and Document collection (selective)	Observation and Simultaneous verbalisation: Note taking and Document collection (selective)	Observation and Simultaneous verbalisation: Note taking and Document collection (selective)	Observation and Simultaneous verbalisation: Video-taping and Document collection ("exhaustive")
	The SE was observed throughout his task of developing the SW for an industrial programmable controller (IPC)	The MDE was observed throughout his task of defining the funct. specs for AMT's computerised control part (IPC)	The SE was observed throughout his task of debugging AMT's computerised control part (IPC)	The CSD was selectively observed during his task of designing an unfurling antenna	The MDE was observed throughout his task of designing a mountain bike attachment
OBSERVATION PERIOD	4 weeks (full time)	3 weeks (full time)	4 weeks (full time)	9 weeks (at a rate of 3-4 days/week)	2 hours

"constructive"[18], and "creative" activities. Stolterman (1992), e.g., opens the *Abstract* of his essay "How system designers think about design and methods. Some reflections based on an interview study" writing: "Today system design methods seem to presuppose the system design process as problem solving, i.e. as repair of a malfunctioning reality". Later on in his paper, he completes this stance, writing: "The design process should instead be viewed as a creative way to design a new reality". In Hartfield's interview with Kelley (1996), Kelley uses the term "problem-solving" in the sense of "fixing something that is deficient or defective", and concludes: "design is not problem-solving" (p. 153) (see also Winograd, 1996, p. xxiii).
In the situativity community (see below), the term "problem" is also the object of many comments. "Situated cognition research calls into question what constitutes a problem for a person…. Problems in cryptarithmetic and the Tower of Hanoi, as 'situations' given to a 'problem solver,' are contrasted with the manner in which a person experiences a problematic situation" writes Clancey (1993, p. 104) (referring, amongst others, to Dewey).

A "problem" in cognitive psychology. These approaches to "problem" and "problem solving", however, actually do not address the classical cognitive psychology approach to these entities. In cognitive psychology, "problem" is a technical term that is precisely defined. Analysed from a cognitive-psychology viewpoint, all that is difficult does not constitute necessarily a "problem". A task does not constitute a "problem" or not *per se*. The "problem" character of a task is a relative feature, depending on the task situation and on the person who is in charge of the task. A task that constitutes a "problem" for one person does not necessarily do so for another one, as has been noticed long time ago already by cognitive ergonomists (Leplat, 1981) (see also the section on "Task vs. Activity" hereunder). "A person is confronted with a problem when he wants something and does not know immediately what series of actions he can perform to get it". (Newell & Simon, 1972, p. 72) This definition is not so far away from the one proposed in 1945 by the psychologist Karl Duncker, foreman of Gestalt psychology, i.e. a rather different approach to cognition than the one represented by Newell and Simon. Duncker considers that "a problem exists when a living organism has a goal but does not know how this goal is to be reached" (p. 2, quoted in Gilhooly, 1989, p. 2). Gilhooly offers an information processing rephrasing of Duncker's definition: "a problem exists when an information-processing system has a goal condition that cannot be satisfied without a search process" (1989, p. 2).
For several authors who adopt the information-processing approach, a "problem" is the representation of a task that does not immediately evoke a "legal" procedure and that allows one to attain the goal (Hoc, 1988; Richard, 1990). This definition leads to the opposition between problem-solving tasks, i.e. tasks that lead to the elaboration of a procedure, and execution tasks in which one applies an existing procedure that has been elaborated previously.
The section on routine and non-routine problems will present Mayer's distinction between these two types of problems. In Mayer's view, a problem may exist when a task "although not eliciting an automatic memorized answer, can be solved by applying a well-known procedure". Such tasks constitute "routine problems".

[18] According to "constructivism", people construct their reality. There is no "objective reality" that would be the same for everybody. People's perceptions depend on their past experiences. "The world is not given as objective forms, rather what we perceive as properties and events is constructed in the context of coordinated activity. Representational forms are constructed and given meaning in a perceptual process, which involves interactivity with the environment, detecting differences and similarities, and hence creating information". (Clancey, 1991) In addition to "constructivist", other adjectives encountered in this context are "constructivistic" and "constructionist".

The relative nature of "problem". Thus, for a particular person, the "problem" character of a task depends on the knowledge this person comes to bring to this task and on the representation that the person constructs of that task. In a problem situation, people cannot retrieve from memory a procedure allowing them to fulfil immediately their task; they first have to solve the problem, leading them to elaborate such a procedure. Thus, a problem-solving task is opposed to a procedure execution task.
This relative nature of the "problem" character of a task can of course be extended to other dimensions than the person confronted with this task and their knowledge. The "importance" or "relevance" of a problem may, e.g. differ according to the company, or the wider society in which it is handled (see Jonas, 1993, who considers "design as problem solving" an appropriate approach if one considers the cognitive process, but a "misleading" view "in the context of society").
NOTE: As mentioned already, authors adopting the information-processing approach to problem solving may have "contributed" to the possible misunderstandings surrounding the notion of "problem". At some occasions, they indeed used the terms "problem forming" or "problem finding" as distinct from "problem solving" —as did Simon in his 1987 lecture delivered to the First International Congress on Planning and Design Theory (1987/1995). The general esprit, however, of the cognitive psychology approach to problem solving is the following. Make part of "problem solving", all activities leading from a problem specification —however vague, imprecise, incomplete it may be, as generally is the case in design projects— to its solution —be it temporary or definitive.

The double status of "problem" and "solution". The *problem* that design starts with, are specifications, generally provided by the client. This problem specifies more or less precisely the artefact to be designed. It has to result in (i.e. has to be solved into) a *solution*, that is, into specifications (in many domains, to the workshop) that specify precisely how to implement the artefact. The problem-solving path, consisting in a transformation from the initial problem specifications into a solution, comprises a great number of intermediary steps leading to a great number of intermediary states, each with a double status. Until the final implementation level is attained, i.e. as long as a "solution" is not yet detailed and concrete enough to specify its implementation, each solution developed for a problem constitutes itself a new problem to be solved. That is why the same entity may be referred to as a "solution" or as a "problem", depending on whether it is an output or an input of a problem-solving action.
Therefore, first, the term "design problem" refers to a designer's starting point for a design project, without any assumption with respect to
- predefined, pre-existing features that discharge the designer from, e.g., constructing a representation of this problem (the problem is no "given");
- the problem representation being constructed once and for all (i.e., at the beginning of the project).

Second, the term "design problem" also refers to later "solution" states elaborated based on these initial specifications. It even refers to all representations of the design project on which the designer continues to work. As long as a designer does not consider the design project as "finished", the design problem is not yet "solved"!
In engineering problems, the client's specifications generally specify the artefact to be designed in terms of a goal to be attained under certain functional and often material constraints: e.g., an automatic installation for manufacturing certain parts at a certain speed (Visser, 1990), or a satellite fulfilling a certain mission with a certain useful charge (Visser, 1991b).
Even if the ultimate solution of the design process consists of the specifications of an artefact —that is, it is not an artefact itself— this solution specifies a *concrete* solution, outlining *how* to *implement* the specifications. Before this ultimate solution is proposed, the global design problem-solving process often proceeds by analysing and further specifying the problem specifica-

tions, in order to transform the goal in terms of a *concept* or a *function* that allows attainment of this goal.
On the level of intermediate sub-problem solving, the same progression may be observed, that is, from a goal, via a concept or a function, to its implementation. At this level however, a problem, defined in terms of a goal to be attained, may be solved immediately in concrete terms —a situation never observed at the global design process level, except in the case of routine design.

"Problem" or "difficulty"□a question of "cognitive cost". In this context, "difficulty" as distinguished from "problem" can be characterised as the amount of processing effort required to solve a problem (Gilhooly, 1989, p. 3). From a cognitive psychology viewpoint, it may be considered a synonym of what we have been calling elsewhere "cognitive cost". "The cognitive cost of an action depends on the number of information units to be processed and the nature of this processing (both the cost of accessing the required information, and that of using it)" (Visser, 1994a) (see also Sperandio, 1980). "Problem" and "difficulty" are thus correlated, but different viewpoints on a task.

"Problem solving"□a global process vs. a particular stage. Thus, as used until now in this subsection, the term "problem solving" refers to a global process, which actually takes place through a big number of small steps, "from problem detection through various attempts to problem solution or problem abandonment" (Gilhooly, 1989, p. 5). These different steps may be characterised as, e.g., "problem forming" or "problem finding", to use two distinctions used by Simon (1987/1995; see also Thomas, 1989), "problem attempting", "problem formulation" (Gilhooly, 1989; Thomas, 1989), "problem setting" (Schön, 1983), "problem conceptualising", "problem structuring", "problem framing" or "reframing" (Schön, 1988), "problem defining" or "redefining".
In our discussion of stage models, we presented a long list of "stages" enumerated by Thomas (1989). Some problem-solving related processes that this list adds to those already mentioned in the previous paragraph are: "idea generation", "idea evaluation", "solution evaluation", "idea integration", "solution acceptance or modification", "goal setting" and "goal elaboration".
In addition to the problem solving methods of heuristic programming (generate-and-test, hill climbing, match, heuristic search, and induction), Newell (1969, p. 407) identifies, as "the many [other] parts of problem solving", recognition, evaluation, representation, information acquisition, method identification, method construction, executive construction and representation construction.

□*ewell and Simon's symbolic information-processing approach to problem solving.* Only a very brief characterisation of the SIP approach to problem solving in general, can be presented in this text (see Simon, 1978; Simon, 1979), but it will be further discussed in the context of the SIP approach to design.
Having qualified a problem as something one wants, but does not know right away how to get, Newell and Simon analyse the problem in terms of "states" and "operators". A problem consists of an "initial state", a "goal state", and of a set of "operators" that are "legal", i.e. that may be used for transforming the initial state into the goal state. Operators have constraints that must be satisfied before they can be applied.
Problem solving takes place in a "problem space" that contains these states, operators, and constraints and that is constructed by the problem solver. Search plays a central role in the SIP approach to problem solving. Confronted with a task, a problem solver "represents the situation in terms of a *problem space*, which is his way of viewing the task environment" (Simon, 1978, p. 272). "The search for a solution is an odyssey through the problem space, from one knowledge state to another, until the current knowledge state includes the problem solution". (Simon, 1978, p. 276)

Several aspects of this approach will be discussed below, especially the roles of search and of representation.

1.6.2 Definitions in the domain of "data collection"

The definitions presented in this section are restricted to a number of questions that we have observed to lead to misunderstandings when we have been presenting our design research in conferences in the domain of HCI and/or Artificial Intelligence and design.

Explorational studies. The appropriateness of different data-collection methods depends on the goal that one has in data collection (hypothesis testing, comparison, evaluation, exploration) and the nature of the activity one is interested in (see Gilmore, 1990).
An *explorational* approach is often used to answer "open" questions, like "How do people X?" or "Which problems are involved by doing Y?". The method may also be called "observation" (e.g., as the first moment in the "experimental cycle" —observation, hypothesis, verification— proposed in 1865 by Claude Bernard in his "Introduction à l'étude de la médecine expérimentale" [Introduction to the experimental study of medical science]). In this text, the term "observation" will be used for referring to one particular type of explorational method, presented below.
As proposed by Claude Bernard, exploration generally is the first phase of research in a new domain —which design still constitutes, even after some 25 years of empirical studies.
"The prime difficulties with exploratory data collection are that it can be very difficult to collect and record such data and it is rare for the resultant data to be well-defined". (Gilmore, 1990, p. 86)

Example. Contrary to a tendency among "situativity theory" proponents, we consider that observation is "guided" by a theoretical framework, even if this framework is only sketchy and temporary[19]. In our first studies of software designers (Morais & Visser, 1985; Visser, 1985, 1986b, 1986c; Visser & Richard, 1984), we did not know the importance of solution reuse. Our framework, at the time, did not yet include the notion of "reuse", so we did not "observe" reuse! We did not code the solutions proposed by the designers as such —even if the designers themselves did refer explicitly to these solutions as "old solutions" or as solutions "already used in the past" (or using a comparable expression).

Because, until now, much research on design has been conducted with the aim of exploring rather than of hypothesis testing, much design research has been explorational, leading to conclusions which are hypotheses for further study.
Two particular explorational methods will be referred to in this text, i.e. observational studies combined with simultaneous verbalisation, and interviews. With respect to the data-collection methods used in cognitive design studies, more examples and details can be found in (Falzon & Visser, 1989; Visser & Falzon, 1988, 1989; Visser & Morais, 1988; Visser & Morais, 1991)[20].

Observational studies. In observational studies, one generally wants to collect data concerning as much aspects of the activity as possible. In cognitive design studies, one wants to collect both data on the internal and on the external, production, and communication aspects.
In controlled studies of small problems, the whole problem solving session can be videotaped (Cross et al., 1997; Cross et al., 1996). In field studies, however, conducted on industrial design

[19] Bucciarelli, however, declares that "what one sees as designing… depends upon one's interests and perspective" (1988, p. 159)
[20] Darses, Détienne, Falzon and Visser (September 2001) present a method for analysing collective design processes, integrating protocol analysis as developed for the analysis of individually conducted activities (Ericsson & Simon, 1980; Ericsson & Simon, 1984) and pragmatic linguistics' verbal-interaction analysis (Kerbrat-Orecchioni, 1990, 1992, 1994).

projects that take weeks, if not months, such exhaustive data collection is generally not realistic —it can be done: the data can be collected, but their analysis would be unfeasible. Therefore, in that case, researchers often take notes on designers' actions, and collect all documents that designers produce during their work.

NOTE: In a field study focusing on a particular component or aspect of the design activity, one may of course videotape selectively. This requires, however, that one already have a kind of model of the global activity so that one knows when this particular component or aspect intervenes in the global process.

The following presentation is based on our data collection in an observational study conducted on design of mechanical functional specifications (the Functional specification study, see Table 1).

□*otes* may concern:

- The designers' problem-solving actions (disclosed by their verbalisation, see below), their other remarks and comments;
- The order in which they produce the different documents, and how they gradually build them up;
- The changes they make;
- The information sources consulted;
- The events considered by the observer to be indicators of the designers encountering difficulties.

Documents collected may be:

- The different versions of the plans, schemas and other external design representations, both the intermediate and definitive versions, and both the versions for the client and those designers construct for themselves during problem solving;
- Photocopies of all documents consulted.

This approach was adopted in our three early design studies (Software design, Functional specification, and Software debugging, see Table 1).

□*erbalisation vs. Introspection*. In addition to note taking and document collection, researchers often ask the people they observe to "verbalise their thoughts" or to "think aloud", i.e., they ask them to report all mental activity with respect to the information they take into consideration, the choices they are confronted with, the criteria used to take a decision, their reasoning, their hesitations, their questioning past decisions, etc. (Ericsson & Simon, 1980; Ericsson & Simon, 1984).

Traditionally, there is an essential difference between this "(concurrent) verbalisation" and the "introspection" method used around the century —and still often used by people developing methods or assistance tools (working in the domain of design methodology, software engineering or HCI).

Introspection consists in commenting *upon* one's mental activity, that is not verbalising this activity, but talking *about* it. What is asked to be formulated in studies applying verbalisation —that may be simultaneous or retrospective— is the information attended to that has been generated by the task-directed processes —in so far as it may be made explicit, which is not always possible. If researchers are interested in a mental activity, e.g., the problem-solving activity underlying a design activity, they do not want to collect comments, translating opinions about, or rationalisations of, the mental activity in question. The data they want to obtain are the *direct traces* of the information used in the mental activity, which are indirect traces of the internal cognitive processes underlying the activity. In order to be used in descriptions or models of the cognitive activity in question, these data is to be analysed, according to strict rules and methods. This analysis constitutes a necessary, subsequent step that cannot be replaced by "subjects [speculating] and [theorising] about their processes…. [One wants to leave] the theory-building part of the enterprise to the experimenter. There is no reason to suppose that the subjects them-

selves will or can be aware of the limitations of the data they are providing [when probed to proceed to introspection]". (Ericsson & Simon, 1980, p. 221)
The use of verbalisation is submitted to several conditions. The two most important ones are expressed by the following summary that Ericsson and Simon give of the effects that the instruction to verbalise may have on the cognitive process that the researcher is interested in.
"The results of [the systematic studies investigating these effects] consistently support our model's prediction that producing verbal reports of information directly available in propositional [i.e., language or verbal] form does not change the course and structure of the cognitive processes. However, instructions that require subjects to recode information in order to report it may affect these processes". (Ericsson & Simon, 1980, p. 235)
"Only information in focal attention can be verbalized" (ibid.). "Automation…. [makes] the intermediate products [of task-directed information processing] unavailable to STM [short-term memory], hence unavailable for verbal reports". (Ericsson & Simon, 1980, p. 225)
Thus, according to the nature of the target activity, verbalisation is a more or less appropriate method. It would, for example, not be useful to obtain data concerning the knowledge used by native speakers in their language activities, because this knowledge is automated.
The technique generally adopted by cognitive psychologists in order to analyse such verbalisations is "(verbal) protocol analysis" (Ericsson & Simon, 1980; Ericsson & Simon, 1984)[21]. Ericsson and Simon's arguments that, if the conditions are satisfied, protocol data can be collected during task execution without interfering with task performance, has been criticised (see the famous, always referred to, Nisbett & Wilson, 1977 paper).
Verbalisation and protocol analysis is often used in "clinical" studies, that is, in analysis of only a few, or even only one single subject, with ensuing possible risks concerning possible generalisation of the results. As claimed, however, by Anzai and Simon (1979, p. 136) with respect to this type of work, "[if] one swallow does not make a summer,… one swallow does prove the existence of swallows. And careful dissection of even one swallow may provide a great deal of reliable information about swallow anatomy".
Since the early days of research on design activities (Eastman, 1970), many, if not most, empirical design studies, especially the cognitive ones, use simultaneous verbalisation (Akin, 1979/1984; Baykan, 1996; Galle, 1996; Gero & McNeill, 1998; Letovsky, 1986; McNeill, Gero, & Warren, 1998; Suwa & Tversky, 1997).
In our studies examining planning and organisation of the design activity, we have shown the risks of introspection (Visser, 1990). We have shown indeed that the plans presented by designers as guiding their activity do not coincide with the actual organisation of their activity. Designers presented these plans as representing the organisation of their activity. Therefore, it would be questionable to use these descriptions based on introspection as data representing the actual organisation of these designers' activity (see also Malhotra et al., 1980).
NOTE. Recently, other data-collection methods are being proposed that might be qualified as situated "between introspection and verbalisation" (Vermersch in Depraz, Varela, & Vermersch, 2003; Vermersch, 1994).

ask vs. Activity. Before presenting *Interviews*, the couple composed of "task" and "(cognitive) activity" needs comment.
The concept "task" refers to, either what people are supposed to do (i.e. their "prescribed" task, as it has been specified by their manager, by instructions or manuals), or the task that they set to themselves and that they actually realise (their "actual" task). Only seldom, these two coincide (Leplat, 1981).
"Cognitive activity" refers to the way people actually realise their task on a cognitive level: the way they make use of knowledge and other information sources they refer to (and of other tools), their use of them in elaborating their intermediary and final productions.

[21] See Note 19, for the corresponding analysis of collective design situations.

The difference between task and activity involves that interviews may provide data on somebody's "task", but not on many aspects of their activity that are relevant for understanding this activity.
In the Interviews section below, an example will be presented to illustrate the importance of the difference between "actual task" (based on data collected in interviews) and "activity" (based on data collected by real-time observations).

Interviews. The remarks formulated in the *Verbalisation vs. Introspection* section with respect to the data "verbalisation" or "introspection" may provide —or may not— on mental activities, apply, *mutatis mutandis*, to the data that one might obtain with interview-like methods. These are the types of methods often used in order to develop interfaces or other interactive system components.
Several types of interviews are possible, from the completely "open", undirected interview, which is nearly like a conversation, to the completely structured interview, around a particular type of questions. The completely undirected interview will rarely be used in a research context, because, even if researchers do not know which data they are going to obtain, they generally know the type of data to be collected, so they will structure their interview around certain types of questions.
Semi-directed interviews are generally the first approach used in the study of a "new" domain of research. It allows data to be obtained on important aspects of the activity, such as a global description of the task (see below), its major stages or phases, the main information sources used.
However, this method must be used with circumspection. It risks inducing the interviewees to present a much more structured representation of their activity than the actual structure of the activity to which they (think they) refer. The following example illustrates the importance of the difference between "actual task" and "activity".

Example. At the start of a longitudinal study, we interviewed designers about their "activity"; afterwards we observed their actual activity. On the basis of the data concerning this actual activity, we concluded that the structure of the representation that the designers presented in the interviews as representing their activity did in fact not coincide with the structure of the actual activity (Visser, 1991a). The designers described their activity as being organised in a hierarchical way, but their actual activity was opportunistically organised —a characteristic of design activities that, since these early days, has been observed more and more in empirical design studies (Visser & Hoc, 1990).

2 The classical symbolic information-processing (SIP) approach to design

One of the first and main authors who have contributed to establish the fundaments of a design theory from a cognitive viewpoint is Herbert A. Simon —who besides has analysed design also from several other viewpoints.

NOTE: For many researchers, Simon is not especially renowned as an author in the domain of design. According to one's discipline, one knows Simon as the winner of the 1978 Nobel Prize in economics, or as an author in Artificial Intelligence. For many psychologists, Simon is, together with Newell, the founder of the main cognitivistic approach to problem solving, i.e. the symbolic information processing (SIP) approach (Newell & Simon, 1972)[22]. It were also Newell and Simon together who received, in 1975, the "Nobel prize" in computer science, i.e. the Turing Award (Newell & Simon, 1976).

For Simon, 1955 and 1956 were the most important years of his scientific life. At the end of 1955, his attention moved from administration and economy to psychology and computer science. This shift was due to the discovery he made together with Newell and Cliff Shaw that the computer could be used as a machine to manipulate symbols (Pitrat, 2002, p. 14) (see Newell, Shaw, & Simon, 1957a, 1957b). In 1958, they published their first cognitive-psychology paper in the *Psychological Review* (Newell, Shaw, & Simon, 1958). The year 1956 is often considered the birth date of Artificial Intelligence, at the Dartmouth conference organised by John McCarthy and Marvin Minsky. Pitrat (2002, p. 14) notices that the participants to this conference mainly presented ideas and projects, but that Newell and Simon came with the Logic Theorist, a system that functioned, and could solve problems that were non-trivial for humans.

2.1 Outline of this section

This section presents the classical symbolic information-processing (SIP) approach to design, mainly based on Simon's work. After several forms and/or formulations of Simon's definition of design, it presents Simon's approach to design from two viewpoints. It first discusses how Simon considered one of the characteristics of design often cited as its main feature, i.e. the "ill-structured" character of design problems. It then introduces three contributions that Simon has made to the analysis of design as an activity and that are still widely recognised as central to design. These are design as a type of cognitive activity rather than as a professional status, design as a problem-solving activity, and design as "satisficing".

Nuances and critiques with respect to this classical cognitive-science position will be presented in Section 4, because they contribute to characterise our approach to design, which is the object of Section 4.

2.2 Simon's research on design

Contrary to Simon's contribution to a general theory of problem solving, his work on design was analytical. With one or two exceptions (Kim, Javier-Lerch, & Simon, 1995), Simon indeed did not conduct any empirical study on design. From the end of the fifties on, however, he has been realising, in collaboration with different colleagues, a considerable body of research on scientific discovery (more than 40 papers —(Cagan, Kotovsky, & Simon, 2001; Klahr & Simon, 2001; Kulkarni & Simon, 1988; Okada & Simon, 1997; Qin & Simon, 1990; Simon, 1977a, 1992b, 1992c, 2001b)— and two books —(Langley, Simon, Bradshaw, & Zytkow, 1987; Simon, 1977a)). In our view, scientific discovery involves the same cognitive activities as those

[22] In his 1995 IJCAI Award for Research Excellence lecture, Simon (1995) declares that his "interest in AI has been, from the beginning, primarily an interest in its application to psychology" (p. 939). Newell and Simon meet in 1952, at the Rand Corporation: "it was love at first sight at the intellectual level" (Pitrat, 2002, p. 14) "Quickly they realised that they shared a common view on the functioning of the brain, symbol manipulator for Simon and information processing for Newell". (Pitrat, 2002, p. 14) Their two first joint papers were (Simon & Newell, 1956) and (Newell & Simon, 1956).

implemented in design, but Simon does nearly establish no link with design (see, however, Cagan et al., 2001).
Except for one study (Okada & Simon, 1997), as far as we know, Simon was concerned with individually conducted design problem solving —as he was in his other problem solving research. This does not mean that he underestimated the importance of collective problem solving. In the sixties and seventies of the twentieth century, few psychologists handled with collectively conducted activities, analysed from a cognitive viewpoint —there was of course research in social psychology, but these studies didn't deal with cognitive aspects of problem solving.
Simon, whose bibliography comprises nearly 1000 titles, among which some 700 papers published in journals in domains ranging from public management to the axiomatisation of physical theories (see http://www.psy.cmu.edu/psy/faculty/hsimon/HSBib-1970.html retrieved May 14, 2003), only published some ten papers directly concerned with design (Cagan et al., 2001; Kim et al., 1995; Simon, 1971/1975, 1973/1984; Simon, 1977b, 1980; Simon, 1987/1995; Simon, 1997; Simon, 1999). The number amounts to some 20 if one also includes publications handling mostly with organisational design, but that do not discuss cognitive aspects of this type of design. These are publications on "The business school: A problem in organizational design" (Simon, 1967), "Information storage as a problem in organization design" (Simon, 1970) or "Programs in public policy: Issues in organizational design" (Simon & Crecine, 1982).
Sciences of the artificial (Simon, 1999) is, however, one of Simon's seminal works and one of the definitely fundamental references exploited in cognitive analyses of design. So is Simon's paper on *The structure of ill-structured problems* (1973/1984). They are two of Simon's central publications in his work on design and will be two of the main references in the rest of this section.

2.3 Simon's framework for design: Sciences of the artificial

"Sciences of the artificial" is the qualification that Simon proposes in 1969 for the "science(s) of design". The nature of design is indeed central in the entire book, even if only two chapters are dedicated specifically to the subject.
The book went into three editions, each one revised. Its first, i.e. 1969 edition, introduced the chapter "The science of design: creating the artificial". In 1981, a revised version of the original collection of chapters constituted a second edition of the book, which also introduced a second chapter specifically on design, i.e. "Social planning: designing the evolving artifact" and two other new chapters. Taken together, the conclusions of the two design chapters constitute the main lines of a curriculum for design education formulated by Simon.
The first edition collected the three Compton lectures that Simon gave in 1968, as well as a 1962 paper on "The architecture of complexity: hierarchic systems". In the Compton lectures, Simon developed his thesis of the contingency of artificial phenomena that had been central to much of his research, at first in organisation theory, next in economics and management science, and later in psychology. The second edition alternated revised versions of the three Compton chapters with the three Gaither lectures delivered by Simon in 1980. In 1996, the third edition revised the previous one and introduced a new chapter on complexity, "Alternative views of complexity".
In the present text, the page numbers for quotations from *Sciences of the artificial* come from the third printing of the third edition of the book (Simon, 1999).
In *Sciences of the artificial*, Simon considers the sciences of design as sciences in their own right. He sees them as distinct from the natural sciences that are traditionally considered as "science". Yet, in 1987, Simon proposes to "compromise" on a perhaps less "pretentious" qualification, speaking of "the art and science of design" (Simon, 1987/1995, p. 245)[23]. As Simon writes

[23] Simon's (1987/1995) paper referred to here, is the text of a lecture delivered in 1987 at the *First International Congress on Planning and Design Theory*. In this text, Simon presents an approach to design that is

in the chapter entitled "The science of design: creating the artificial" (in which engineering design is the reference), "Historically and traditionally, it has been the task of the science disciplines to teach about natural things: how they are and how they work. It has been the task of engineering schools to teach about artificial things: how to make artifacts that have desired properties and how to design". (Simon, 1999, p. 111) The natural sciences are concerned with the necessary, with how things are, whereas design is concerned with the contingent, with how things might be (Simon, 1999, p. xii). Designers are "concerned with how things *ought* to be… in order to *attain goals* and to *function*" (Simon, 1999, pp. 4-5). Simon's thesis is indeed that "certain phenomena are 'artificial' in a very specific sense: they are as they are only because of a system's being molded, by goals or purposes, to the environment in which it lives" (Simon, 1999, p. xi). That is why symbol systems (or "information processing systems") are "almost the quintessential artifacts[:] adaptivity to an environment is their whole *raison d'□tre*" (Simon, 1999, p. 22).

2.3.1 Two steps in Simon□s contribution to a SIP design theory

Two steps can be distinguished in Simon's contribution to a cognitive design theory. The first one was taken with Newell, to whom "Sciences of the artificial" is being dedicated "in memory of a friendship". Together, the two researchers elaborated what has since been called the principles underlying the "symbolic information-processing" approach to problem solving (Newell & Simon, 1972) —or abridged the "symbolic processing" (Greeno & Moore, 1993, pp. 57-58), "symbolic" (Vera & Simon, 1993a, p. 10), or "information-processing" approach (Simon, 1978 □2151, p. 272) (here abridged as the "SIP" approach). It is also frequently referred to —often by authors adopting a different approach— as the "traditional", "computational" view. The SIP approach has been one of the main starting points of the "cognitivistic" perspective in cognitive science. For some 20 years, it has been the theoretical reference for the cognitive analysis, not only of problem solving (Miller, Galanter, & Pribram, 1960; Reitman, 1965), but also of other types of activity: concept learning (Bruner, Goodnow, & Austin, 1956), verbal understanding and memory (Anderson, 1983, 1976; Le Ny, 1989a, 1989b; Le Ny, 1979). Together with various colleagues, Newell and Simon also used this general approach in order to explore broader domains than the one analysed in their famous 1972 *Human problem solving* (1972). They analysed concept formation, verbal learning, and perception, but also administrative and organisational behaviour, creativity and scientific discovery, and even music and emotion (for references, see Newell & Simon, 1972, p. 791, Note 1).

It was Simon who, subsequently, applied this paradigm to design (Simon, 1971/1975, 1973/1984, 1987/1995, 1999). In these analyses of design, Simon identified and elaborated various characteristics of this specific problem solving activity that have formed the basis of the approach taken toward design by many, if not most, researchers in cognitive psychology and cognitive ergonomics who have been conducting research on design since the early eighties. These characteristics will be presented below, after a short section presenting different attitudes adopted by different authors with respect to the framework proposed by Simon for analysing design.

2.3.2 Different attitudes with respect to Simon□s design framework

Eastman (1969) was one of the first researchers to adopt the SIP framework for the analysis of design. He did so in a particularly original, interesting precursor work in the domain of empirical design research. Since Eastman, many authors, both in cognitive psychology and cognitive ergonomics, and in other disciplines, globally adopt Simon's framework, occasionally proposing

much closer to ours than is the one presented in *Sciences of the artificial*. We will come back to this 1987 lecture later on in our text. One of our "questions to Simon" (Visser, 2002b) concerns our surprise that Simon's more "flexible" position expressed in 1987 (or perhaps only in 1995) was not reflected in his 1996 version of the "The science of design" chapter.

a few complements or modifications (Akin, 1986; Baykan, 1996; Goel, 1994; Goel & Pirolli, 1992).
Other authors, from psychology, but mostly from other disciplines —i.e. sociology, ethnology, and anthropology—, propose what they consider a completely different, new paradigm to design studies (Bucciarelli, 1988; Bucciarelli & Sharville, 1996; Bucciarelli, 1984; Schön, 1983, 1988, 1992, March). Focusing on the so-called "situativity" approaches, these alternatives will be discussed in Section 3.
The position adopted in this text is
- is to pay a tribute to Simon for his fundamental contribution to design theory,
- but also to consider that Simon's SIP approach
  - is to be reformulated on several, essential points, some of which may be found in the "situativity" approaches, and
  - is to be completed with features that characterise actual design as implemented in professional design tasks.

2.4 Simon's definition of design

Simon provided various definitions and characterisations of design. Some general definitions translating Simon's vision concerning the essential aspects of design, are presented here. Other, more specific views will be presented and discussed in later sections.
In *Sciences of the artificial*, Simon proposes that "everyone designs who devises courses of action aimed at changing existing situations into preferred ones" (1999, p. 111). A "design as problem solving" definition is: "design is a planned activity, directed toward a goal or set of goals, and subject to an evolving set of constraints". (Cagan et al., 2001, p. 453)
Closer to design methodology and engineering viewpoints is Simon's following definition. "Design… means synthesis. It means conceiving of objects, of processes, of ideas for accomplishing goals, and showing how these objects, processes, or ideas can be realised. Design is the complement of analysis —for analysis means understanding the properties and implications of an object, process, or idea that has already been conceived". (1987/1995, p. 246) (cf. the ASE paradigm)
Finally, one may recognise the information processing and Artificial Intelligence modelling approach in the last definition quoted in this sub-section. "Design is inherently computational —a matter of computing the implications of initial assumptions and combinations of them. An omniscient God has no need to design: the outcome is known before the process starts. To design is to gather information about what follows from what one has proposed or assumed. It is of interest only to creatures of limited information and limited computing power —creatures of bounded rationality like ourselves". Simon, 1987/1995, p. 247]
Simon's view of design activity as *problem solving* in the standard information-processing sense of *search* in a problem space, is one of the characteristics of his approach that is going to evoke much debate —as in our approach to design.

□□□

The following presentation of Simon's approach to design exposes the design characteristics that Simon has identified and that are still central in today's cognitive design research. This presentation has two parts, one presenting Simon characterising design problems, and the other Simon's approach to design as an activity.

2.5 Design problems: "ill-structured" problems

A characteristic of design problems nowadays considered as one of its main specificities, is their ill-defined, or in Simon's terms "ill-structured" character.
Simon (1973/1984) starts his famous paper on "The structure of ill-structured problems" by discussing the impossibility to compose a formal definition of "well-structured problems" (WSPs). He therefore dresses instead "a list of requirements that have been proposed at one time or another as criteria a problem must satisfy in order to be regarded as well-structured" (Simon,

1973/1984, p. 182). "A problem may be regarded as well-structured to the extent that it has some or all of [these] characteristics". (Simon, 1973/1984, p. 183)
Reasoning from the example of designing a house, presented as a creative architectural design problem, Simon (1973/1984) lists some characteristics making it an "ill-structured problem" (ISP). At the outset, when designers start to work on a design brief, and even after having constructed a representation of the design project, their data do not fulfil the criteria for well-structured problems. What, especially, lacks are some of the requirements "that have been proposed at one time or another as criteria a problem must satisfy in order to be regarded as well-structured":

- a definite criterion for testing any proposed solution, not to speak of a mechanisable process for applying the eventual criterion;
- one or more problem spaces in which can be represented the initial problem state, the goal state, and all other, intermediary states that may be reached, or considered, in the course of attempting a solution to the problem;
- one or more problem spaces in which can be represented any knowledge that the problem solver can acquire about the problem;
- a possibility to define with complete accuracy the changes in the world that the design may bring about.

Simon does not consider "ill-structuredness" or "well-structuredness" an absolute characteristic. "The boundary between well-structured and ill-structured problems is vague, fluid and not susceptible to formalization" (Simon, 1973/1984, p. 181). "In fact, many kinds of problems often treated as well-structured are better regarded as ill-structured" (p. 182). Examples of such problems that are "often treated as well-structured", but that might be "better regarded as ill-structured" are chess playing and theorem proving, two types of problems on which are based the analyses underlying Newell and Simon's SIP model (1972). Nevertheless, this does not lead Simon to conclude that for human problem solvers there are no well-structured problems, and that every "real", or even realistic, problem is ill-structured. According to Simon, one must rather reconsider the nature of ill-structured problems. In his view, it is only at first analysis that "typically" "ill-structured" problems, such as creative design problems, may be considered ill-structured. Reasoning on the example cited above (i.e., design of a house), Simon argues that even such a "typically ill-structured" problem rapidly acquires structure, due to a designer applying certain strategies, such as decomposition. "General problem solving mechanisms that have shown themselves to be efficacious for handling large, albeit *apparently* well-structured domains should be extendable to ill-structured domains without any need for introducing qualitatively new components". (Simon, 1973/1984, p. 197)
"During any given short period of time, the architect will find himself working on a problem which, perhaps beginning in an ill-structured state, soon converts itself through evocation from memory into a well-structured problem". (Simon, 1973/1984, p. 190) Both chess playing and architectural design problems, even if they are ill-structured "in the large", are well-structured "in the small" (Simon, 1973/1984, p. 191). "In the small" refers to the sub-problems to be solved (in chess, playing one single move; in architecture, e.g., designing a heating system or another component of a house to be designed), "in the large" to the global problem (in chess, a complete game; in architecture, the entire project).
Thus, for Simon, design problems may seem ill-structured at first sight, but the designer rapidly structures them and then solves the well-structured problem version! This also holds for apparently very "well" structured problems, such as algebra-like puzzles, symbolic logic, or the famous Tower of Hanoi problems.
This means that, if researchers refer to Simon for their characterisation of design problem as "ill defined" or "ill structured", they misrepresent Simon's view. Simon was indeed one of the main authors at the origin of the discussion of design problems as "ill defined" or "ill structured", but for him, ill structuredness was not a particularity of design problems.

2.6 Design activity

Three essential characteristics of design identified by Simon concern design as an activity. The central one is the vision of design as a symbolic information-processing problem-solving activity. The two other ones are, however, also important and are the characteristics still prevalently adopted in many cognitive design studies —even those conducted by researchers who don't espouse the SIP framework.

2.6.1 Design as a type of cognitive activity rather than as a professional status

For ergonomists, Simon is an intellectual precursor when he states, in 1969, that "design" is not restricted to engineers. "Engineers are not the only professional designers. Everyone designs who devises courses of action aimed at changing existing situations into preferred ones. The intellectual activity that produces material artifacts is no different fundamentally from the one that prescribes remedies for a sick patient or the one that devises a new sales plan for a company or a social welfare policy for a state" (Simon, 1999, p. 111). Engineering, medicine, business, education, law, architecture, and painting are "all centrally concerned with the process of design" (ibid.).

Nowadays, the view that design is a type of cognitive activity, not a professional status restricted to certain professionals, called "designers", is adopted by many, if not most, researchers in cognitive psychology and cognitive ergonomics of design —even if they do not always adopt as large a circumscription of activities "centrally concerned with the process of design" as Simon does.

Cognitive psychologists obviously agree on the approach of an activity in terms of the kind of mental representations and processes that are implemented, rather than in terms of the status of the person implementing the activity —be it their socioprofessional or a different status.

2.6.2 Design as "satisficing"

A second, significant design characteristic, which constrains people to make decisions without complete information, is design's "satisficing" quality. This very general concept —not at all restricted to design activity— appears in one of Simon's first reports for the Rand Corporation, where he became a consultant in 1952 (according to Pitrat, 2002, p. 13).

Satisficing is to "settle for the good enough" (Simon, 1971/1975, p. 1). It is the alternative to "optimising". Satisficing consists in finding an acceptable, satisfactory solution (Simon, 1987/1995, p. 246) rather than calculating the optimum value, or choosing the best solution amongst all possibilities. "Seldom will the goals and constraints be satisfied by only a single, unique design; and seldom will it be feasible to examine all possible designs to decide which one is, in some sense, optimal". (Simon, 1987/1995, p. 246).

This need for satisficing is mainly due to human "bounded rationality" (Simon, 1982, 1997). Bounded rationality is a typical human characteristic due to people's severely limited abilities with respect both to knowledge and information holding and accessing, and to computing power, i.e. people's "memory contents and their processes" (Simon, 2000, p. 25)[24]. This Simon's approach to design —and other human activities— as a "satisficing" activity has been widely adopted as central to design in cognitive design studies.

With respect to the selection criteria that may be used, Simon, in one of his rare papers on design (Simon, 1971/1975), analyses "style" —in a very broad, not only aesthetic sense— as being one of the factors entering in "choosing any one of many satisfactory solutions".

Anticipating our discussion of Simon's position, we may note the following. In the distinction between generation and evaluation activities in design (an opposition that Simon does not establish —not explicitly, in any case), Simon's "satisficing" view concerns evaluation rather than

[24] See "operation research", a field in Computer Science that is concerned with finding as fast as possible an as good as possible solution to complex problems. The philosophy of operation research is that finding a reasonable solution in a reasonable time is better than finding the optimal solution in an infinite time.

generation. For Simon, design alternatives are "found", but the way in which they are found is not discussed. "Once we have found a candidate we can ask: 'Does the alternative satisfy all the design criteria?'" (Simon, 1999, p. 121) Because it is rarely feasible to examine all possible designs in order to choose the optimal one, designs are evaluated by comparing them in terms of "better" and "worse", more or less "acceptable", more or less "satisfactory" ("satisficing"). Dealing with trade-offs among different possibilities, among different constraints that are to be satisfied, plays a central role in this satisficing activity, a design characteristic that we will discuss when presenting our approach to design.

2.6.3 Design as a symbolic information processing problem solving activity

As briefly mentioned above, it was in their famous *Human problem solving* book from 1972, that Newell and Simon (1972) proposed to analyse problem solving as a symbolic information processing activity (SIP). From the late fifties, early sixties of the twentieth century on, this approach had been developed by the authors, together with colleagues such as Shaw (Newell et al., 1958; for other references, see Newell & Simon, 1972, p. 791, Note 1) (for a succinct presentation of the SIP approach to problem solving, see e.g. Simon, 1978; Simon, 1979).

Design as "problem solving". For Simon, the central defining characteristic of the design activity is its problem solving nature —even if "creative design" is not a "common-place" problem-solving activity (Simon, 2001a, p. 205 and p. 206).
The application of the SIP paradigm to design was not made right from the start (i.e. in Simon et al.'s studies conducted during the nineteen-sixties, or in Newell & Simon, 1972). This was not due to any theoretical reason, but neither design, nor other "real life", let alone professional, tasks were being studied by Simon and colleagues. Newell and Simon (1972) note that their work is concerned with tasks that are short (half-hour), and that concern moderately difficult problems of a symbolic nature resolved by "intelligent adults" working individually. The elaboration, since the fifties, of the 1972 SIP model had been based indeed on algebra-like puzzles (cryptarithmetic), chess, and symbolic logic problems —that we will refer to as the "classical laboratory problems" in the rest of this text. Newell and Simon (1972) note that the thinking involved in, e.g., "designing a house, discovering a new scientific law,… creating new music… are largely beyond the current state of the art", i.e. as it was around 1970 (p. 7). Nevertheless, "they are in fact part of the same story [the authors] wish to tell" (ibid.). In later years, together with colleagues and Ph.D. students, Simon began tackling such problems, especially scientific discovery (Kulkarni & Simon, 1988; Langley et al., 1987; Okada & Simon, 1997; Qin & Simon, 1990; Simon, 1992b, 1992c).
In the early years of cognitive psychology, many authors embraced the SIP paradigm as the fundamental schema for their investigation of problem solving (e.g. Miller et al., 1960). Already in 1964, Reitman (1964), e.g., adopted a representation for problem solving that could be formalised using the IPL-V information processing language elaborated by Newell, Shaw, Simon and other colleagues in the nineteen-sixties[25]. Reitman applied this problem-solving schema to the solving of what he qualified as "ill-defined" problems (see below).
Simon asserts that no new and hitherto unknown concepts or techniques are needed to handle design problems or ill-structured problems. "No qualitatively new components" need to be introduced in the classic general problem solving mechanisms in order to handle these problems (Simon, 1973/1984, p. 197), no "special logic" is necessary in order to deal with design problems (Simon, 1999, p. 115) —even if Simon "admits" that standard logic is to be adapted to the search for alternatives, i.e. solution elements (Simon, 1999, p. 124). This is one of the various examples of Simon applying his "nothing special" position (presented in Klahr & Simon, 2001, p. 76, for scientific thinking).

[25] IPL (Information-Processing Language) was the first list-processing computer language.

Solving ill-structured problems in two consecutive stages: first structuring, then solving. As we saw, for Simon, an ill-structured global problem —such as the design of a house— is rather smoothly amenable to a set of well-structured sub-problems. Solving ill-structured problems (IPS), for Simon, indeed proceeds through two consecutive stages, first structuring the ill-structured problem, then solving the resulting well-structured problem or problems (WSP). Simon's only remark with respect to the relative importance of this structuring activity is the following. "There is merit to the claim that much problem solving effort is directed at structuring problems, and only a fraction of it at solving problems once they are structured". (Simon, 1973/1984, p. 187) If one realises how much time in product-development projects, is spent in design —and especially in the early "conceptual design" stages (Sharpe, 1995), i.e. the core of design activity from a cognitive viewpoint—, this "structuring" activity is then perhaps what characterises these early development stages.

In his comments and answers in reaction to questions formulated at the conference *Sciences de l'Intelligence, Sciences de l'Artificiel* [Sciences of the Intelligence, Sciences of the Artificial] (La Grande-Motte, Fr., 1-4 Feb. 1984), Simon discusses if "problem formulation" can be considered "problem solving". He concludes that the processes that lead from a problem situation to its solution are the same as those leading from a situation to the formulation of a problem (Simon, 1984/2002, p. 44). He refers to his paper "The structure of ill-structured problems" (Simon, 1973/1984) as a first approach to the analysis of these processes. Simon is optimistic with respect to the possibility to model the activity involved in solving poorly structured problems. He bases this optimism in the following reasoning. To solve an ill-structured problem requires the formulation of goals —as do all goal-oriented activities, such as problem solving. Even if the goals of an ill-structured problem are imprecise, they necessarily rely on criteria of which the person already disposes —be it implicitly. Therefore, according to Simon (1984/2002), rather than to consider that goals are to be "formulated", it would be more appropriately to characterise the activity as "evocation" and "definition" of goals.

2.7 Conclusion

This section has presented both the contributions of the SIP approach to design that we consider "positive", and some of its specificities that we will critique in a later section, where they will serve the exposition of our own approach to design. The "positive" contributions are Simon's analysis of design as a type of cognitive activity rather than a professional status, as "satisficing", and as problem solving. Simon's contribution to the "design problems as ill-structured" discussion has played an important role in later cognitive design studies. We do not concur, however, with his viewpoint on this question.

In the first part of the section presenting our approach to design, we will discuss the SIP approach to design, identifying six points on which this approach, in our opinion, misrepresents design. We will also contrast this discussion with Simon's "more nuanced" positions in more recent texts.

3 Alternatives to the symbolic information processing approach. Focus on the "situativity" approach

The traditional approach to cognition as applied to design, discussed in the previous section has been severely criticised in recent years. Some of these criticisms are based, in our view, on terminological confusions, such as those already referred to above, especially concerning "problem" and "problem solving". We will see that an important role in the discussion is also played by some notions not yet introduced above, especially "symbol" and "situated action" —that is, the central concepts on both sides (given that we will focus on the situativity approach as alternative to SIP). Certain criticisms that the new approaches have been addressing to the SIP approach seem justified to me —and are related to criticisms that I am also formulating here.
This text is —as far as we know— one of the first to outline and discuss in somewhat detail these criticisms, confronting them with the SIP positions, mostly based on references and quotations from Simon's work. That is why this section is rather detailed.

3.1 Outline of this section

This section briefly opens and closes with some views on human cognitive activities proposed as alternatives to the SIP approach. Cognitive ergonomics in France is positioned with respect to the situativity approach. The section further presents two explicitly conducted "Symbolic Information Processing" vs. "Situativity" debates, focussing on the famous 1993 Special issue of *Cognitive Science* on Situated action. It then discusses the situativity approach to design as mainly represented by Donald Schön.

3.2 Alternative views on human cognitive activities

From the 1980s on, several researchers have started to formulate, with respect to the SIP approach, alternative views on human activities in their actual accomplishment —or have been showing renewed interest in older theories, especially in "activity theory" and in that proposed by Piaget. "Situated cognition" and "situated action" are the main labels put on these new views, by the authors themselves or by their opponents. A related approach, analyzing situations in terms of "distributed cognition", will not be further discussed in this text, amongst other reasons because it seems more related to collective than to individual design (Hollan, Hutchins, & Kirsh, June 2000; Hutchins, 1995; Hutchins, 1996).
For mainly two reasons, we will focus in this text on "situativity theory" —rather than on "distributed cognition" and "activity theory". First, among these different approaches, situativity is best representing the authors proposing an alternative perspective to the analysis of design (whose main representatives are Schön, Bucciarelli and Gero). Second, there have been explicit debates between this family of approaches and SIP. For a particularly clear comparison of activity theory, situativity theory, and distributed cognition, see Nardi (1996b), who makes "an argument… for the advantages of activity theory as an overall framework" (p. 70).
NOTE: Greeno and Moore (1993, p. 50), propose to use the term "situativity theory" "rather than the term in more common use *theory of situated cognition*". They consider that "the phrase *situated cognition* often is interpreted, understandably, as meaning a kind of cognition that is different from cognition that is not situated" whereas the authors consider situativity "a general characteristic of cognition". We adopt Greeno and Moore's terminological suggestion.
Authors frequently referred to in the context of situativity theory and who are among the forerunners at the source of these approaches are Winograd (1986), Suchman (1990, reprint of the 1st Ed., 1987), Agre (1987), Lave (1988), Brown (1989), and Clancey (1991).
Many sympathisers of the situativity approach are working in the teaching and learning community[26] (e.g. Greeno, 1998, January/February 1997; Greeno & Moore, 1993; Lave & Wenger,

[26] As shown, e.g., by pages proposed in reaction to a Google search for "situated" (conducted on 26/06/02).

1991) (see also the Educational Researcher debate presented below). Indeed, even if most of the new perspectives present a rather general view on cognition, they often have been used especially in certain application domains —and, symmetrically, certain application domains seem to be the privileged domain of particular theoretical perspectives.

3.3 Cognitive ergonomics in France and situativity

Cognitive ergonomics, in any case as "practised" in France[27], is inherently "situated" —even if not in the sense of the "classical" situativity approach. It has been thus since its origin and so embedded is this viewpoint that most researchers in the domain until recently did not even feel the need to use this qualification explicitly.

French cognitive ergonomics is "situated" in that one of its main premises is that the characteristics of the situations in which people are working are quintessential for the understanding and modelling —and modification— of these people's situations. Relying heavily on "work analysis"[28], it mainly conducts its studies in people's actual workplaces, observing and examining people while involved in their actual work and their actual working conditions. Referring to Leontiev, forerunner of activity theory, French ergonomists introduced the distinction between "task" and "activity" (see above, the section "Definitions in the domain of data collection") —a distinction that continues to be essential in French ergonomic research (Desnoyers & Daniellou, 1989; see also the collection of papers in Leplat, 1992a; Leplat, 1992b).

In the conclusion of their presentation of the SELF, the Francophone Ergonomics Society, Desnoyers and Daniellou (1989) write: "Although Francophone Ergonomists have insisted on the specificity of their approach to work, they are quite aware and interested in convergent approaches developed elsewhere…. Cognitive Anthropology, Ethnographic Description, Situation Awareness, Situated Action are… domains Francophone Ergonomists feel close to.

This convergence has led some of us to reconsider the use of the expression Francophone Ergonomics De Montmollin [1995] has proposed that we should refer to this type of Ergonomics under the name of Ergonomics of Activity The new denomination has the advantage of describing the field in a more functional manner, and it does better convey the message that Francophone Ergonomists are looking for more frequent and more intense exchanges with their colleagues throughout the world".

3.4 "Symbolic Information Processing" SIP vs. "Situativity" SIT debates

Several publications have been the place of a "Symbolic Information Processing" vs. "Situated Cognition" / "Situated Action" (SA) debate, especially the famous 1993 Special issue of *Cognitive Science* on Situated action ("Special issue on Situated action," 1993) and the discussion in the *Educational esearcher* between, on one side, Anderson, Reder and Simon (Anderson, Reder, & Simon, 1996; January/February 1997) and, on the other, Greeno (January/February 1997). In this text, we will concentrate on the first debate, because the second one focuses on "Situated learning and education" (Anderson et al., 1996), a theme not particularly central to our present preoccupations. The structure of the *Educational esearcher* discussion merits, however, a short presentation.

Anderson, Reder and Simon (1996) argue that "advocates of a situated cognition approach to learning and teaching often overstate the empirical support for their position and underestimate the empirical support for rival points of view", as characterised by the *Educational esearcher* editor, Donmoyer (1996, May). According to the three authors, "much of what is claimed by [the 'situated learning' movement] is not 'theoretically sound'" (Anderson et al., 1996, p. 5). In their paper, Anderson, Reder, and Simon review what they consider "the four central claims of situated learning with respect to education":

[27] Desnoyers and Daniellou refer to "Francophone" ergonomics (Desnoyers & Daniellou, 1989).

[28] "Work analysis can be described as addressing three phases: the description of the actual work situation, that of the activity of the operator, that of the effects of this activity both on the operator and the work system". (Desnoyers & Daniellou, 1989)

"action is grounded in the concrete situation in which it occurs;
knowledge does not transfer between tasks;
training in abstraction is of little use; and
instruction must be done in complex, social environments.
In each case, [they] cite empirical literature to show that the claims are overstated and that some of the educational implications that have been taken from these claims are misguided". (Anderson et al., 1996, p. 5)
In his response to this critique to the situated perspective on learning, Greeno (January/February 1997) first notes that the four propositions contested by Anderson et al. are incorrectly called "claims of situated learning" by these authors. Second, he considers that the empirical evidence that Anderson et al. advance in order to show the defects of situativity, is, to the contrary, "compatible with the framing assumptions of situativity; therefore, deciding between the perspectives will involve broader considerations than those presented in [Anderson et al.'s] article". (Greeno, January/February 1997, p. 5)
As we will see, this situation is typical of the SIP - SIT debate: the proponents of each approach present what they consider "evidence" that shows the defects of the other approach. In response, the authors defending this other approach refute the "evidence", or its relevance.
The *Cognitive Science* Special issue "contains a debate among proponents of two distinct approaches to the study of human cognition. One approach, the tradition upon which cognitive science was founded, is that of symbolic processing, represented in the article by Alonso Vera and Herbert Simon [which triggered the debate]. The other more recent approach, emphasizing the role of the environment, the context, the social and cultural setting, and the situations in which actors find themselves, is variously called situated action or situation cognition. It is represented by four articles written in response to Vera and Simon: James Greeno and Joyce Moore; Philip Agre; Lucy Suchman; and William Clancey". (Norman, 1993, p. 1)
Among the topics that are central in these discussions, a certain number of questions that are most relevant to our text are structuring their discussion in what follows. "What is at issue?" is the first one. "What is SA?" receives not only an answer by its proponents, but also by its opponents —entertaining or even leading to several disagreements. "What is a symbol?" plays a similar role. In order to discuss these questions, we have identified two couples of critical distinctions: denotation vs. interpretation and recognition vs. direct perception.

3.4.1 □ hat is at issue□

After a short recapitulation of the nature of "physical symbol systems", Vera and Simon (1993a) start by discussing their view of SA and its claims. They contest these claims and present symbolic systems that, in their opinion, perform well in the situations requiring SA according to the situativity authors, "whereas the systems usually regarded as exemplifying SA are thoroughly symbolic (and representational)" (p. 7).
For Greeno and Moore (1993) the question "seems to be something like this: whether (1) to treat cognition that involves symbols as a special case of cognitive activity, or (2) to treat situated activity as a special case of cognitive activity, with the assumption that symbolic processing is fundamental in all cognitive activity. [Greeno and Moore] advocate the first option; Vera and Simon advocate the second". (p. 50)
For Agre, "the critical issue is whether one's categories locate things in agents and worlds separately [as does the 'symbolic worldview' represented by Vera and Simon] or in the relationships between them" as do the SA authors —even if they all "provide different accounts of what the latter might mean" (Agre, 1993, p. 69).
For Clancey (1993), there is a "category error" in play. "The symbolic approach conflates 'first-person' representations in our environment (e.g., utterances and drawings) with 'third-person' representations (e.g., mappings a neurobiologist finds between sensory surfaces and neural structures…)". (pp. 87-88) The symbolic approach "[ignores] the shift of reference between the agent and the scientist looking inside" (Clancey, 1993, p. 88). "Symbolic models have explana-

tory value as psychological descriptions.… But arguing that all behavior can be *represented* as symbolic models [as do Vera and Simon] misses the point: We can model everything symbolically. However, what is the residue? What can people do that computer programs cannot? What remains to be replicated?" (p. 89)

3.4.2 □ hat is a "symbol"□

For Vera and Simon, "symbols are patterns", which means that "pairs of them can be compared" (Vera & Simon, 1993a, p. 9). Their physical nature is "irrelevant to their role in behavior" (ibid.). They can be "of any kind: numerical, verbal, visual or auditory" (Simon, 2001a, p. 205). "Symbolic", thus, "is not synonymous with 'verbal'" (Vera & Simon, 1993a, p. 10). Crucial is that "we call patterns symbols when they can designate or denote" (Vera & Simon, 1993a, p. 9).
In order to act in an environment, people use perceptual and motor processes that connect the "symbol system" that is built from symbols, with its environment. "A… symbol system is built from… symbols, which may be formed into symbol structures by means of a set of relations". (Vera & Simon, 1993a, p. 8) "A symbol system has a memory capable of storing and retaining symbols and symbol structures, and has a set of information processes that form symbol structures as a function of sensory stimuli, which produce symbol structures that cause motor actions and modify symbol structures in memory in a variety of ways". (Vera & Simon, 1993a, p. 9)
The perceptual and motor processes that connect the symbol system with its environment provide the symbol system "with its semantics, the operational definitions of its symbols. Evocation of a symbol by stimuli emanating from the thing or situation it designates also provides the system with access to (some or all of) the information stored in memory about the thing designated. The memory is an indexed encyclopedia". (Vera & Simon, 1993a, pp. 9-10)
For Greeno and Moore, "a symbol or symbolic expression is a structure —physical or mental— that is interpreted as a representation of something" (1993, p. 50). The authors consider that the distinction between symbolic and non-symbolic processes depends on "whether a process includes a semantic interpretation of a symbolic expression, that is, an interpretation that gives the symbolic expression referential meaning". (p. 54)
According to Greeno and Moore, their use of the term *symbol* is "consistent with a long tradition in philosophy, psychology, and linguistics. Examples include Dewey's (1938) distinction between *signs* and *symbols*, and Peirce's (1902/1955) distinction between *indices* and *symbols*". (p. 50) The authors consider that the definitions used by their opponents in the debate (i.e. Vera and Simon) are inappropriate.
For Vera and Simon, however, Greeno and Moore's is "an alternative, albeit rather restrictive definition of symbols, but it is not the one that cognitive scientists working within the symbolic-system hypothesis have used" (Vera & Simon, 1993b, p. 78). In their short "Reply to reviewers", Vera and Simon don't quote names of any of these cognitive scientists. An example of a French researcher is Le Ny (1989a), who in his research on verbal understanding indeed adopts the same stand as Vera and Simon. For this view of "symbols", Le Ny refers to formal semantics and to Peirce "interpreted in a carnapian and tarskian vision".
We will come back to this opposition concerning denotation and interpretation.
Agre considers Vera and Simon's definitions of "symbols" and "symbol systems" as "metaphors". For this SIT author, "there exist many reasonable uses of 'symbol', each embedded within a worldview" (1993, p. 64). According to Agre, "Vera and Simon fail to distinguish between worldviews and theories. A 'worldview' is a largely unarticulated system of vocabulary, methods, and values shared by a research community. Suchman's (1987) worldview is ethnomethodological; Lave's (1988) is dialectical; mine might be called interactionist. A 'theory' is some substantive proposition formulated within a worldview. You argue against a theory by presenting contrary evidence. But a given worldview will admit a wide variety of alternative theories". (Agre, 1993, p. 62)
Vera and Simon, in their opening paper, had discussed Agre and Chapman (1987)'s video game program Pengi as one of the "current research projects that have been viewed as exemplifying

SA principles" (Vera & Simon, 1993a, p. 24). For them, Pengi "satisfies the definition of a symbol system" (Vera & Simon, 1993a, p. 37): it "[seems] to have categorical representations of states in the world and functional characterizations of those states" (ibid.).
Agre (1993) presents Pengi as having arisen "from a dissatisfaction with STRIPS-like planners and with world models" (p. 67). Completely in line with his analysis of Vera and Simon's positions as a "worldview", Agre replies in the following terms. "Whether Pengi's hardware can be reconstructed in 'symbolic' terms is beside the point; the point is that we interpret it in interactionist terms, and we hold that these are the best terms for analyzing the phenomena". (Agre, 1993, p. 68)

3.4.3 □ hat is "situativity"□

As for "symbol", the different authors participating in the debate adopt different definitions of this other central notion —even within the SIT movement (see Agre, 1993's observation quoted above).

□*era and Simon's view*. For the two authors defending the SIP approach, a sensible use of the term "situated action" refers to action that occurs "in the face of severe real-time requirements" and that is "based on rather meager representations of the situation" (Vera & Simon, 1993a, p. 41). Vera and Simon judge that all action requires "some internal representation of the situation —perhaps minimal in the case of situated action, more elaborate in the case of planned behavior when fewer unexpected events occur" (ibid.).
The authors consider that they can give a symbolic interpretation of SA. "Some past and present symbolic systems are SA systems. The symbolic systems appropriate to tasks calling for situated action do, however, have special characteristics that are interesting in their own right". (p. 8) "The term 'situated action' can best serve as a name for those symbolic systems that are specifically designated to operate adaptively in real time in complex environments". (Vera & Simon, 1993a, p. 47) Vera and Simon even judge that "the goals set forth by the proponents of SA can be attained only within the framework of symbolic systems" (p. 7)

□*reeno and* □*oore's view*. Vera and Simon's opponents in the special issue clearly do not agree. For the authors who have proposed the term "situativity" (Greeno & Moore, 1993), the corresponding theory is a large movement that refers to "emerging scientific practices, empirical findings, and theory [that] include the development of ecological psychology, the ethnographic study of activity, and philosophical situation theory (1993, p. 49).
The opinion attributed by Vera and Simon (1993a, pp. 7-8) to SA proponents, according to which SA researchers deny that "symbolic processing lies at the heart of intelligence" is indeed endorsed by Greeno and Moore (1993, p. 50). However, as noted by these SA authors, "the issue hinges crucially on the meaning and theoretical status of the concept of *symbol*" (Greeno & Moore, 1993, p. 49). "The central claim of situativity theory is that cognitive activities should be understood primarily as interactions between agents and physical systems and with other people. Symbols are often important parts of the situations that people interact with [but] we expect that accounts of most —perhaps all— individual and social cognitive phenomena will include hypotheses about processes that are not symbolic". (Greeno & Moore, 1993, pp. 49-50)

3.4.4 Two critical distinctions: denotation vs. interpretation and recognition vs. direct perception

"Symbols" and "situativity" are both viewed in different ways by the authors from both sides. "Symbol", however, seems to be the main element of controversy (see, e.g., Greeno and Moore's remark quoted in the previous paragraph).
As we saw above, for Vera and Simon, symbols "designate or denote" (1993a, p. 9), whereas for Greeno and Moore, "symbolic expressions" are structures that are "interpreted as [representations] of something" (1993, p. 50).

The distinction between "designation / denotation" and "interpretation" seems essential in this debate. It is related to another crucial distinction, the one between "recognition" and "direct perception" —itself related to the conscious and/or verbalisable character of processing.
According to Newell and Simon, (1972), "a symbol structure *designates* (equivalently, *references* or *points to*) an object if there exists information processes that admit the symbol structure as input and either:
(a) affect the object; or
(b) produce, as output, symbol structures that depend on the object". (p. 21)
"The relation between a designating symbol and the object it points to can have any degree of directness or indirectness". (ibid.)
For Vera and Simon, on the one hand, "symbolic theories generally make no specific assumptions about what part of the processing takes place at a conscious level and what part is unconscious, except that symbols held in short-term memory (in the focus of attention) are generally available to consciousness, and often can be reported verbally". (1993a, p. 10). On the other hand, "awareness has nothing to do with whether something is represented symbolically, or in some other way, or not at all. It has to do with whether or not particular symbols are available to consciousness in short-term memory. Thus, in an act of recognition, the symbol denoting the object recognized is consciously available, the symbols denoting the features that led to the recognition generally are not. The recognizer is aware of the former but not of the latter". (Vera & Simon, 1993a, p. 19) "Only a small part of denotation entails direct perception of objects and their behavior, as the denotation of 'cat' mainly does. The connections between a symbol structure and its denotation can be complex and highly indirect (e.g., the denotations of concepts like 'empirically true,' or even 'China')". (Vera & Simon, 1993a, p. 12) In the same way, as Vera and Simon (1993c, p. 128) notice in their reply to Clancey (1993), the "rules" used in a computer system that simulates human activity are not necessarily accessible to, nor verbalisable by, the person whose activity is being simulated.
Thus, in Vera and Simon's view of "symbolic processing", many activities, cognitive and other, are not conscious, and *yet* proceed by underlying symbolic processes. In Greeno and Moore's view of "symbolic processing", many activities, cognitive and other, are not conscious, *and consequently do not proceed* by underlying symbolic processes.
NOTE: This opposition is not merely terminological. Greeno and Moore use the terms "direct perception" and "recognition" as distinctive notions (for whose meaning they refer to Gibson, 1966, and Neisser, 1989, 1992). They consider "recognition" a symbolic process, as do Vera and Simon, but "direct perception" is no symbolic process for Greeno and Moore. For Vera and Simon, "recognition" is a technical term with a precise definition. We do not know, however, if the authors' use of "direct perception" (in the quotation given above) also conveys such a particular, technical sense. If it does, it does not seem to be the same as for their opponents in the debate. An assertion by Vera and Simon (1993b) in their *reply to reviewers* supports our belief to this respect. The authors indeed affirm that "the problem with creating a distinction between direct perception and recognition is that there seems to be no principled way of deciding what sorts of phenomena fall into which category". (p. 79)
So, when Greeno and Moore state that "there is more than 'symbolic processing'", one can only agree —providing that one adopts the authors' definition of "symbols". Indeed, not all processing requires conscious, semantic, interpretational activities!
Simon considers that "the currently popular ideas of situated action and situated learning" have the following "simple essence". "In those frequent cases in everyday experience in which attention is directed primarily to the external environment, the cues in this constantly changing scene, which are constantly modified by the actions of the system, provide the triggers for evoking the successive productions". (1992a, p. 132) It is through "recognition, in which cues permit the recognition of familiar situations or objects and thereby gain access to the information about them that is stored in memory" (id., p. 132)

"A principal reason why [many unrigorous procedures are] satisfactory is that people are in constant interaction with their environment. We take a step, notice new conditions and constraints that our reasoning ignored, and adjust our next step accordingly. In designing, we record our tentative design decisions in a drawing, and the drawing reveals interactions that were ignored in our premises.…. Here again, is situated action arising from the operation of a production system". (id. p. 134)

3.5 Situativity and design

In the writings by the authors who have applied a situated perspective to design, one does not find such a precise discussion of the central notions of situativity as presented in the discussions introduced above. The absence of precision and/or the vagueness of the presentations (descriptions, characterisations) of design are precisely among our main reserves with respect to the situativity approach.

Schön and Bucciarelli were, as far as we know, the first researchers applying this approach to design. Rather than "situated", Schön uses terms such as "reflective practice" and "knowing-in-action" and "reflection-in-action", that we interpret as forms of what the situativity authors have called "situated action" and "situated cognition". Both authors have studied design mostly in collective settings. Bucciarelli particularly focused on collective design from a social perspective. Schön presented what he considered general characteristics of design, even if displayed in, and even due to, the communicative context of the design situation. That is why Schön's work will be presented in most detail. A third author presented below is Gero, who in recent work, also gives a central role to the situativity notion in his analysis of architectural design.

3.5.1 Sch□n

In a presentation of "Donald Alan Schön (1930-1997)" in □*he Encyclopedia of Informal Education* (July 2001), Smith writes that even if Schön was "trained as a philosopher,… it was his concern with the development of reflective practice and learning systems within organizations and communities for which he is remembered". Indeed, in design circles, one generally refers to Schön as the author who, through his proposal of the reflective-practice concept, offered an alternative to the SIP approach defended by Simon in □*he sciences of the artificial* (Simon, 1999). Schön's research and thoughts on design thus originate from an educational perspective[29]. His enterprise is concerned with the way in which "*professionals think in action*" as "*reflective practicioner*s" (Schön, 1983), and the consequent *education* of such reflective practicioners (Schön, 1987a, 1987b). Referring, for his phrase "*reflective conversation with the situation*", to Dewey (who also provides Schön with the expression of people being in "transaction" with a situation), Schön considers "designing as reflective conversation with the materials of a design situation" (title of his 1992, March paper).

Relative to the contrast between "reflection-in-action" and "school knowledge" (1987a), Schön does not see himself "as saying anything really new at all". He is drawing on "a tradition of reform and criticism which begins with Rousseau and goes on to Pestilotsy and Tolstoy and Dewey and then, as we approach more contemporary times, Alfred Schultz and Lev Vygotsky and Kurt Lewin, Piaget, Wittgenstein and David Hawkins today" (1987a)[30]. Reflection-in-action is the "kind of artistry that good teachers in their everyday work often display", whereas "school knowledge" refers to "the view that what we know is a product", that "the more general and the more theoretical the knowledge, the higher it is", and to the notion that "knowledge is molecu-

[29] Schön was an educator. Having "recieved his B.A. from Yale University, and an M.A. and a Ph.D. from Harvard University, all of them in philosophy", he was "Ford Professor Emeritus on Urban Studies and Education, and Senior Lecturer in the Department of Urban Studies and Planning, and Architecture, at the Massachusetts Institute of Technology, the institution that he joined in the early 70's until his death in 1997" (Pakman, 2000, p. 5).

[30] See the discussion on "situated learning", briefly presented above, between Anderson, Reder and Simon (Anderson et al., 1996; January/February 1997) and Greeno (January/February 1997).

lar". From the school-knowledge perspective, "it is the business of kids to get it, and of the teachers to see that they get it". (Schön, 1987a)

Schön's studies of design have indeed focused on design students learning with experienced designers (Schön, 1992, March; Schön & Wiggins, 1992, Spring). These studies have been conducted in "reflective practicums such as the design studio in architecture" (Schön, 1987a) (see also Schön, 1984).

Proceeding as an ethnologist or anthropologist (see Suchman, 1990, reprint of the 1st Ed., 1987) in his analysis of particular work situations, Schön (1983) discusses such specific situations in detail in order to reveal the central role of *reflection-in-action* in professionals' practice. He concludes that in their "reflective conversation, the practitioner's effort to solve the reframed problem yields new discoveries which call for new reflection-in-action. The process spirals through stages of appreciation, action, and reappreciation. The unique and uncertain situation comes to be understood through the attempt to change it". (Schön, 1983)

"Design knowledge is knowledge in action" (Schön, 1992, March, p. 3). In order to show the nature of the corresponding *knowing-in-action*, Schön (1987a) uses the example of what happens when you are "riding a bicycle, and you begin to fall to the left". People who *know* riding a bicycle will do the right thing when *in situ*, but will often give the wrong answer when asked certain questions, in a classroom or anywhere else, outside of a bike-riding situation. An example of such a question might be: "If you are riding a bicycle, and you begin to fall to the left, then in order not to fall you must turn your wheel to the □□□?". This contrast between "[doing] the right thing when *in situ*" and being unable to answer correctly when not, requires an explanation. "This capacity to do the right thing,… exhibiting the more that we know in what we do by the way in which we do it, is what I mean by *knowing-in-action*. And this capacity to respond to surprise through improvisation on the spot is what I mean by *reflection-in-action*. When a teacher turns her attention to giving kids reason to listening what they say, then teaching itself becomes a form of reflection-in action, and I think this formulation helps to describe what it is that constitutes teaching".

In his famous book □*he reflective practitioner*□ *How professionals think in action* (Schön, 1983), design is one of a series of activities involving "reflective practice". Architectural design was the first professional domain studied by Schön in order to develop his epistemology of professional practice based on the concept of knowledge-in-action. Examples of other practices examined come from domains such as city planning, engineering, management, and law, but also from education, psychotherapy, and medicine. In his 1983 book, Schön offers "a sample of vignettes of practice, concentrating on episodes in which a senior practitioner tries to help a junior one learn to do something.… The heart of this study is an analysis of the distinctive structure of reflection-in-action".

In one of his first papers handling specifically with design (1988), Schön announces that, "in this paper, [he] will treat designing not primarily as a form of 'problem solving', 'information processing', or 'search', but as a kind of *making*. On this view, design knowledge and reasoning are expressed in designers' transactions with materials, artifacts made, conditions under which they are made, and manner of making". (p. 182) Designing is "a kind of *making*…. What designers make… are *representations* of things to be built". (ibid.)

Schön emphasises that "problem solving" is generally considered as handling problems as "given" (as Simon, e.g., indeed typically does), whereas the process of "problem setting" is neglected. "Problems of choice or decision are solved through the selection, from available means, of the one best suited to established ends. But with this emphasis on problem solving, we ignore problem setting, the process by which we define the decision to be made, the ends to be achieved, and the means that may be chosen. In real-world practice, problems do not present themselves to the practitioner as givens. They must be constructed from the materials of problematic situations which are puzzling, troubling, and uncertain" states Schön (1983, pp. 39-40).

NOTE. For Winograd (1996) neither, design is no problem solving. "Design is creative", "[it] cannot even be comprehended as *problem solving*" (p. xxiii). To this respect, design is opposed to engineering. Engineers are problem solvers, no designers.

For Schön, *problem setting* in architectural design occurs, for example, when architects see the design project in a different, new way, e.g. when they see a T-form figure as two L-form figures back to back. For Schön, his observations and his approach to these observations "should be contrasted with the familiar image of designing as 'search within a problem space'…. The designer *constructs* the design world within which he/she sets the dimensions of his/her problem space, and invents the moves by which he/she attempts to find solutions". (Schön, 1992, March, p. 11).
Another design characteristic, introduced through an example from architectural design, is the series of actions "seeing-moving-seeing" applied on "snippets" (Schön & Wiggins, 1992, Spring). These are actions such as observing a drawing, transforming it, and observing the result, and may lead to the "discovery of certain unintended consequences" of the transformation move (p. 139). Even if architects have a certain intention in transforming a drawing, they are not aware of all possible consequences of their actions. Their intention is liable to evolve in their "conversation" with the drawing. Referring to Simon, Schön notices that it is because of our "limited awareness" and our "limited ability to manage complexity" that designing has "the conversational structure of seeing-moving-seeing" (Schön & Wiggins, 1992, Spring, p. 143). Schön and Wiggins refer several times to *Sciences of the artificial*, in which Simon introduced his idea of limited information processing capacity into the theory of designing. They emphasise, e.g., that "because of our limited information processing capacity, we cannot, in advance of making a particular move, consider all the consequences and qualities we may eventually consider relevant to its evaluation" (1992, Spring, p. 143).
Schön also notices "the remarkable ability of humans to recognize more in the consequences of their moves than they have expected or described ahead of time" (Schön, 1992, March, p. 7). As pointed out, already long ago, by the urban designer Christopher Alexander quoted by Schön, "our ability to recognize qualities of a spatial configuration does not depend on our being able to give a symbolic description of the rules on the basis of which we recognize them" (Schön, 1992, March, p. 137). Analogously, and as also noticed by Christopher Alexander, even if designers are able to make, tacitly, "qualitative judgments", they are not necessarily able to state, that is to make explicit, the criteria based on which they make them (Schön, 1992, March, p. 138).

3.5.2 □ucciarelli

The engineer Bucciarelli (who has collaborated with his colleague the philosopher, as noticed by Schön, 1992, March) presents the method he uses as "participant observation". He "[pretends] to be an ethnographer while participating as an engineer" in engineering design projects (1984, p. 185). Using this approach, Bucciarelli examines the "reflective practice in engineering design" (title of his 1984 paper).
Adopting this "ethnographic perspective on engineering design" (title of his 1988 paper), Bucciarelli considers that "designing is more than a cognitive process, although design knowledge and designers' heuristics are essential ingredients" (1988). He focuses on the collective nature of design, his main premise being that "designing is a social process" —using exactly the same formulation as Schön (1988, p. 182)[31]. "Engineering design is the business of a collective or team [whose] different participants, with different competencies, responsibilities and interest

[31] For "An Annotated Bibliography of Books Related to the Social Study of Design", see Peter L. Kantor's website at http://www.daaq.net/bibliography/index.html (Retrieved November 20, 2003). One of the books and articles reviewed by Kantor is *Discovering Design□Explorations in Design Studies*, "a collection of essays that grew out of a conference on the social aspects of design held in Chicago in 1990. The basic premise of this conference was that design 'cannot be adequately understood apart from the issues and concerns of contemporary cultural discourse' (Buchanan & Margolin, 1995, p. ix)".

'see' the object of design differently" (Bucciarelli, 2002, p. 219). Design is "a process of negociating among disciplines" (Bucciarelli, 1988, p. 160, p. 164). The design that results from such a process is a "social construction" (Bucciarelli, 1988, p. 167). Indeed, "a single object can be understood differently within different 'object-worlds'" (Bucciarelli, 1988, p. 161). In a recent paper, Bucciarelli (2002) describes his vision as follows: "Different participants, with different competencies, skills, responsibilities and interests, inhabit different [object-worlds]. As such, while admittedly working on the same object of design, they see the object differently" (p. 220). In this 2002 paper, the author presents a "tower of babel" vision of design process, which might lead people to "wonder how it succeeds.… Simply put, in many instances design does not succeed". (Ibid.) "But most failures of process are hidden from view, kept within the memory of the collective, never revealed to the world, and even within the firm, often easily forgotten. Few participants would deny that designing can be better done; the process improved". (p. 221)

3.5.3 Gero

Gero is an architect and researcher who has played an important, instigative role in the domain of "Artificial Intelligence in Design" (Gero, 1991; Gero, 1992). For several years now, Gero has been developing his so-called "function-behavior-structure" model (Gero, 1990). More recently, in addition to references to Schön's "reflection in action" and to other authors (cited below), he also uses situativity notions, analysing, e.g., "conceptual designing as a sequence of situated acts" (Gero, 1998).

In his recent texts, Gero (1998; 1998) distinguishes three "models of designing": "designing as search", "designing as planning", and "designing as exploration". "Designing as search" is represented by many artificial intelligence authors, but also by engineering design methodologists, e.g. Dym (1994). "Designing as planning" is closely related to the previous view. "Designing as exploration" is represented by, e.g., Logan and Smithers (1993).

"The basic and often implicit assumption in designing as search [models] is that the state space of possible designs is defined a priori and is bounded.… The designing processes focus on means of traversing this state space to locate either an appropriate or the most appropriate solution (depending on how the problem is formulated).… The assumption that the space is defined prior to searching relegates this model to detail or routine designing". (Gero, 1998, p. 3)

"Designing as exploration takes the view that the state space of possible designs to be searched is not necessarily available at the outset of the design process.… Exploration may be viewed in two ways. It may be viewed as a form of meta-search: the designer searches for state spaces amongst the set of possible predefined state spaces. It may be viewed as a form of construction where each new state space bears some connection to the previously constructed state space(s).… Exploration connects with the ideas of conceptual or non-routine designing". (Gero, 1998, p. 3; see also Logan & Smithers, 1993)

According to Gero's recent view, these three types of models do not adequately model design. This failure has led him to "develop models which include such concepts as reflection and emergence and processes which match those of exploration". (Gero, 1998, p. 4) In order to realise these claims, the author introduces ideas from the "areas" of "situatedness" and "constructive memory" (for which he refers to Rosenfield, 1988, but see also Note 9).

3.6 Situativity and planning

For at least three reasons, the relations between situativity and planning constitute the object of an independent sub-section:

- the book *Plans and situated actions* by Suchman (1990, reprint of the 1st Ed., 1987), one of the forerunners of the situativity approach, gives planning a fundamental role in discussions and comments on situativity;
- in one of its forms, planning is itself a design activity (see the section "Temporal and spatial constraints in route plan design");

- planning and the actual organisation of design are central in our own research (presented below).

Referring to Suchman's book, authors have often viewed the notion of "situated action" as *opposed to* planning. In her "Response to Vera and Simon", Suchman (1993) declares that she doesn't agree with such an interpretation of her approach, but states that the relation between plans and "the actions they project constitutes a central, unanswered question for existing accounts of practical reasoning and action" (p. 75). "Planning is itself a form of situated action… that results in projections that bear some interesting, and as yet unexplicated, relation to the actions they project". (Suchman, 1993, p. 72) Plans are, however, in Suchman's view, not the "control structures" they have often been considered to be —in Artificial Intelligence research, we should want to add—, i.e. they "are not determining… the actions they project, at least not in the strong sense of 'determining'". (Suchman, 1993, p. 74)

In a critical discussion of Suchman's position, Bardram (1997) shows how "plans, however, do play an essential role in realising work". (p. 17) Based on experiences from designing a computer system that supports the collaboration within a hospital, the author discusses "how plans themselves are made out of situated action, and in return are realised in situ". (ibid.) This leads him to propose that "work can be characterised as situated planning" (ibid.). Bardram places this approach to work and planning in the framework of Activity Theory, which emphasises, according to the author, "the connection between plans and the contextual conditions for realising these plans in actual work". (ibid.)

"Suchman (1987) shows the importance of differentiating between work and representations of work like plans and process models. Plans are representations of situated actions produced in the course of action and therefore they become resources for the work rather than they in any strong sense determine its course. Suchman emphasises action as essential situated and *ad hoc* improvisations, which consequently make plans rational anticipations, before the act, and *post hoc* reconstructions, afterward. The theoretical work on situated action, and the studies underlying it, seems to have attained so much attention that the importance of plans and protocols as guidance of work has been neglected. Recently, at the CSCW'96 conference in Boston, Suchman[32] herself commented that an unfortunate, but typical, misreading of her work was that plans do not exist. Plans do exist and should be viewed as 'an artifact of our *reasoning about* action, not... the generative *mechanism* of action.' (p. 39, emphasis in original)". (Bardram, 1997, p. 18)

"Nevertheless, in medical work, *pre-hoc* representations of work like plans, checklists, schedules, protocols, work programmes etc. have proved extremely valuable as mechanisms giving order to work". (Bardram, 1997) Comparable observations are made by "Schmidt and Simone (1996) [who] raise the rhetoric question to Suchman of 'What is it that makes plans such as production schedules, office procedures, classification schemes, etc. useful in the first place? What makes them ‘resources’?' (p. 169)". (Bardram, 1997, p. 18)

In our work on planning and actually organising design activity, we have shown that designers *possess* plans (i.e., as one of their knowledge sources), and *use* them as resources for organising their activity. These plans are, however, not their *only* resource for action, and are indeed not the control structure. The actual organisation of their activity depends not only on these plans, but also on other opportunities-providing knowledge sources (see below, the section "The opportunistic organisation of design").

3.7 Other alternative approaches to design

We are aware of at least one other alternative approach to design. Rittel, the author who together with Webber proposed the notion of "wicked" (vs. "tame" or "benign" problems, Rittel & Webber, 1973/1984, see below) proposes, in a discussion of "planning societal problems", an argumentative model of the design process. One may observe this argumentative structure of the

[32] We suppose that this was in the Panel "From Retrospective to Prospective: The Next Research Agenda for CSCW", in which Suchman was one of the panelists (our Note, WV).

plan design process, if one looks at it as a network of issues, with pros and cons. In an interview that Rittel gave in 1972 on "Second-generation design methods", he states: "Thus the act of designing consists in making up one's mind in favour of, or against, various positions on each issue". (Interviewed by Grant and Protzen, 1972/1984, p. 325) "As distinguished from problems in the natural sciences, which are definable and separable and may have solutions that are findable, the problems of governmental planning —and especially those of social or policy planning— are ill-defined; and they rely upon elusive political judgment for resolution. (Not 'solution'. Social problems are never solved. At best they are only re-solved —over and over again.)" (Rittel & Webber, 1973/1984, p. 136) In the 1972 interview, Rittel declares that one of the two areas of emphasis in further work in design methodology is "the further development and refinement of the argumentative model of the design process, and the study of the logic of the reasoning of the designer [i.e.] the rules of asking questions, generating information, and arriving at judgments.… The second area of emphasis should be work on practical procedures for implementing the argumentative model: the instrumental versions of the model". (pp. 323-324)

3. Conclusion

Donald Schön's approach to design has been presented as the "situativity" approach to design. Indeed, even if Schön's keywords were neither "situated action" nor "situated cognition", his analyses of design situations in terms of "reflective practice", and "knowing-in-action" and "reflection-in-action" seem to us what a "situativity" approach to design might be. Gero who explicitly gives a central position to the "situated" notion refers for his theoretical framework to Schön. The allusive character of this positioning of authors with respect to theoretical paradigms is, in our opinion, typical of the "situativity" approach. This judgment will be further developed in our discussion of the situativity approach in the next section.

4 Our approach to design. Integrating and completing both SIP and SIT approaches

The two previous sections have introduced two families of approaches to design, i.e. the classical SIP and the alternative SIT approaches. The present section presents our approach to design, which aims to complete and integrate these two approaches presented as opposed, but in our view rather complementary.

Our approach has two main grounds. From a theoretical viewpoint, we adopt elements from both SIP and SIT approaches, completing and integrating these two approaches, based on empirical data from cognitive design studies, conducted since some 20 years now. Indeed, our approach is also strongly grounded in the analysis of data collected in empirical design studies, performed mainly in professional work settings, but also in experimental conditions. Much material comes from our own work, that is, mostly empirical studies conducted on professional designers working in different task domains (software, mechanical, and industrial design; see Table 1). Nevertheless, empirical work on designers by other researchers is of course also covered (see references in the section "Some historical pointers") (see also Adelson & Soloway, 1988; Ball & Ormerod, 1995; Bonnardel, 1992; Carroll, Thomas, Miller, & Friedman, 1980; Détienne, 1991b; Guindon, 1990; Guindon, 1990; Rist, 1991a, 1991b; Ullman & Culley, 1994; Whitefield, 1989), to present just a few examples. Software design will receive special attention, as one of the domains in which probably most empirical studies of design have been conducted —and in which we have also been working.

NOTE: From the beginnings of our research on, an important accent has always been on the task - activity opposition and the relevant data-collection methods that go with it, especially real-time observation. This way of data collection has made our research resemble that conducted from a "situativity" standpoint. Always guided, however, by a theoretical framework, our selection and analysis of data did not concur with the esprit of "situativity". At the same time, having adopted at the beginning a SIT perspective, our work deviated more and more from this approach to design problem solving, as we will detail below.

4.1 Outline of this section

This section starts by a discussion of the two main theoretical alternative approaches introduced above. This discussion will focus on aspects of design that are, in our view, misrepresented in these approaches. The discussion of the classical symbolic information-processing approach will also introduce Simon's "more nuanced" positions in more recent texts, which, surprisingly, coexist with his "strict" position (especially as expressed in Simon, 1999).

The proper presentation of our own approach starts by a discussion of general definitions and characterisations of design, mainly from the viewpoint of cognitive psychology and cognitive ergonomics, followed by the proposition of our definition of design. We then present a detailed characterisation of design from two viewpoints, that is, design as a particular type of problem and design as a particular type of activity. Even if the two are intertwined (design activity involves solving design problems and v.v.), it is useful for analytical purposes to distinguish these two forms of characteristics. This section ends by a presentation of two particular forms of design, i.e. design of plans (planning) and software design.

4.2 Discussion of the classical symbolic information processing approach

In the section dedicated to the SIP approach, we have characterised Simon's view of design through four characteristics, (i) one concerning the type of problems (their possible "ill-structuredness"), and three concerning design activity, i.e. (ii) its analysis based on the cognitive activities involved, rather than on the designer's professional status, (iii) its "satisficing" rather than optimising nature, and (iv) its analysis as a symbolic information-processing problem-solving activity. As may be clear from our preceding discussions, we take a different stand than Simon with respect to design problems as "ill-structured " (point i), and to design activity as an

information-processing problem-solving activity (point iv). This last point is related to Simon's analysis of the ill-structured nature of design problems and of their solving as proceeding in two independent, consecutive stages (first structuring the problem, then solving it).
 isrepresentations of design by the classical symbolic information-processing approach. We will address these points in our discussion of other critiques to the SIP approach, especially it underestimating certain aspects of design activity (such as the role of problem representation building and of "nondeterministic" "leaps" in design) and its overestimating other aspects (such as problem decomposition, means-ends analysis, and search). This discussion, organised in six points, concerns design as an activity. The first discussion point relates this activity to the type of problem that design constitutes.
The discussion will focus on the approach proposed by Simon as the researcher who has applied the classical SIP approach to design in *Sciences of the artificial*. We will also refer to Simon's collaborative work with Newell (1972) and Greeno (Greeno & Simon, 1988), and to work by other representatives of the SIP approach, such as Newell's own research (1969).
In general terms, our critique concerns the too systematic, and thereby impoverished, approach to design: Simon represents design as much more orderly than actual design has been shown to be. The essence of Simon's view may well be applicable to rather "simple", well-defined[33] problems and to their processing, but is unrealistic with respect to the typical ill-defined problems that professional designers generally meet.
After this discussion, the section will conclude by a presentation of some of Simon's "more nuanced" positions, mostly expressed in more recent publications, but coexisting in some texts with his "strict" position.

4.2.1 nderestimating the specificity of ill defined problems and their solving

Simon's analysis of the ill-defined character of design problems and its consequences on design problem solving activity, is, as noted already, the first point where I put a different accent than Simon.
As noticed above, Simon (1973/1984) judges that the "general problem solving mechanisms" that are efficacious for well-structured problems, apply just as well to ill-structured problems, i.e. without any need for adaptation. Using decomposition and other "general problem solving mechanisms", designers first transform ill-structured problems into well-structured problems. Subsequently, they solve the resulting well-structured (sub)problems.
An essential point in our analysis of Simon's view on ill-defined problems and their solving, is the discrepancy between his positions with respect to economics and to design thinking. Analysing Simon's ideas on design, we came to wonder why it is that Simon adopts a "subtler" position with respect to economics, than with respect to the psychology of thinking, and especially of design thinking.
With respect to economics, Simon is indeed very sensitive to the way in which its theories — especially neo-classical ones— idealise human rationality, and neglect its limits. It is for this completely new approach that he obtains the Nobel price in economics.
Simon ascribes the idealisation of the human person to these economic theories directing "their attention primarily to the external environment of human thought, to decisions that are optimal for realising the adaptive system's goals". These decisions "would [indeed] be substantively rational in the circumstances defined by the outer environment" (Simon, 1999, p. 23).
In Simon's view, actual human economic behaviour "illustrates well how outer and inner environment interact and, in particular, how an intelligent system's adjustment to its outer environment… is limited by its ability, through knowledge and computation, to discover appropriate adaptive behavior". (Simon, 1999, p. 25)

[33] For reasons exposed below, we use the terms "ill defined" and "well defined", except in discussions of the analyses made by Simon who used the terms "ill structured" and "well structured".

In his 1981 chapter "Economic rationality: adaptive artifice", Simon insists on the often satisficing nature of the economic actor's decisions. "Because real-world optimization, with or without computers, is impossible, the real economic actor is in fact a satisficer, a person who accepts 'good enough' alternatives, not because less is preferred to more but because there is no choice". (Simon, 1999, p. 25)

With respect to human behaviour in another domain of activity, i.e. in situations requiring problem solving[34], however, Simon seems himself to under-estimate certain limitations of humans, both on their rationality and on their attentional capacities.

In the domain of management science, Simon notices that more or less "rational" methods, i.e. methods from operations research to artificial intelligence, "have been applied mainly to business decisions at the middle level of management. A vast range of top management decisions (e.g., strategic decisions about investment, R&D, specialization and diversification, recruitment, development, and retention of managerial talent) is still mostly handled traditionally, that is, by experienced executives' exercise of judgment". (Simon, 1999, p. 28) He adds: "As we will see in [the chapters on the psychology of cognition], so-called 'judgment' turns out to be mainly a non-numerical heuristic search that draws upon information stored in large expert memories". (ibid.)

This under-estimation of design problem-solving complexity may be due, in part, to the types of problems on which the elaboration of the SIP model was based (see above; we will come back on this point in the final Conclusion of this text). Simon may consider design —or what he seems to consider its prototype and default value, i.e. engineering design— as no more complex than those problems whose solution processes can be covered satisfactorily in terms of search through finite (even if not small) problem spaces. In an introductory section of the chapter on "Social planning: designing the evolving artifact", Simon announces that "in the previous chapter representation was discussed mainly in the context of relatively well-structured, middle-sized tasks. Representation problems take on new dimensions where social design is involved". (Simon, 1999, p. 141) This "previous chapter" was concerned with engineering design and architectural design. Simon suggests that differences of at least three types may be involved in the greater "complexity" of social design problems: the more or less well or ill definedness of problems, their size, and the nature of their object. We will come back to this point.

It is with respect to the social planning problem of regulating automobile emission standards that he writes: "One may regard 'defensibility' as a weak standard for a decision on a matter as consequential as automobile emissions. But it is probably the strictest standard we can generally satisfy with real-world problems of this complexity. [Especially in situations of this kind] an appropriate representation of the problem may be essential to organizing efforts toward solution and to achieving some kind of clarity about how proposed solutions are to be judged. Numbers are not the name of this game but rather representational structures that permit functional - reasoning, however qualitative it may be". (Simon, 1999, p. 146) In the section "Designing without final goals" of this "Social planning" chapter, Simon affirms that processes such as "search guided by only the most general heuristics of 'interestingness' or novelty…. may… provide the most suitable model of the social design process". (Simon, 1999, p. 162) He proposed, nevertheless, also this kind of search as providing the mechanism for scientific discovery (in Ch. 4, "Remembering and learning: memory as environment for thought"). Such declarations appearing occasionally in further very straightforward, "simple" or even "simplistic" presentations, may well give us food for thought concerning Simon's genuine position.

Therefore, it seems that only when he discusses economic and/or social problems that Simon takes into consideration the role of what he calls "representations without numbers", generative constraints such as "interestingness" or "novelty", critical constraints such as "defensibility" of a decision, and human's bounded rationality.

[34] Or, if economics also requires problem solving, in situations requiring *other* types of problem solving.

However, relative to Simon's analysis of "typical" design problems, empirical observations on professional designers working on such engineering problems, show that,

- even if well-defined (sub)problems exist, they exist besides many other, ill-defined (sub)problems; and
- even if problems are well-defined at the level of the most concrete sub-problems —i.e. at the level just before implementation specification—, much ill-defined problems will have to be solved before attaining that level;
- however, most importantly, there are not two separable phases in design, i.e., first structuring a problem, and afterwards solving it (see the non "linear" character of the design process referred to in the section on Stage models vs. process models) (This holds for most other types of "real" problem solving). Of course, designers analyse problems, they decompose them, they interpret and reinterpret them, making them thus more manageable, easier to solve. However, such activities occur in an interspersed manner, all through the design process, until late in the project. Many examples of this can be found in empirical studies of strategic aspects of design, especially studies detailing its opportunistic character (Visser, 1994a).

Example (Visser's Functional specification study). This is an example of subproblems remaining long-time ill defined, even after careful problem analysis and decomposition. It concerns the way in which a mechanical engineer (ME) transformed the functional requirements that constituted his "problem" into the functional specifications that constituted the "solution" to this design problem. The ME's requirements were mainly those formulated by the client but, in a first, global analysis, mechanical design colleagues of the ME had already analysed them and had listed specifically the operations that they judged necessary for the required functions. They had done so in a particular specifications document. Nevertheless, this document was not only consulted, but also completed, and even modified, by the ME. The functional requirements indeed admitted several interpretations, thus several solutions (functional specifications) and the ME did not always have the same problem analysis as his colleagues.

As noticed above, the idea of designers proceeding in two main consecutive stages —structuring the ill-structured problem, and solving the resulting well-structured problem— is the prevailing view in Simon's 1973 paper.
Nevertheless, occasionally Simon is less "extreme" and seems to consider most problem solving as switching between ill-defined-problem solving (which nevertheless consists in structuring the ISP into a WSP, and then solving the WSP) and well-defined problem solving (Simon, 1973/1984, p. 197). "Each small phase of the activity appears to be quite well-structured , but the overall process meets none of the criteria we set down for WSPs". (Simon, 1973/1984, p. 194) Problem solvers —designers— can be considered to be "faced at each moment with a well-structured problem, but one that changes from moment to moment" (Simon, 1973/1984, p. 195). Simon, however, does not analyse, or even mention, the probable "reframing" or "redefinition" activity to which designers in that case have to proceed with respect to their design problem. We saw that this type of activities is fundamental according to situativity researchers such as Schön—and so are they in our own approach.
Even if we agree with Simon that "there is no real boundary" between well-defined and ill-defined problems, we think that it has, at least, a clearly heuristic function to distinguish "typically" ill-defined problems (such as design problems) from "typically" well-defined problems (such as the classical laboratory problems). Such a distinction makes it possible, e.g., to anticipate certain particularities of designers' activity when they are confronted with such problems (see below).
To this respect, Newell (1969)'s argumentation may be considered closer to ours than Simon's. In a 1969 paper, Newell defended the position adopted by Simon in his 1973 paper, judging that "typically" well-defined problems such as logic theorem proving, checker playing and pattern

recognition, were nevertheless ill-defined. He advanced, however, as the reason for this position that, in the fifties of the XXth century, for these tasks, "algorithms either did not exist or were so immensely expensive as to preclude their use" (p. 366). Newell discusses Reitman (1964)'s position that such problems may constitute a "small and particularly 'well-formed' subset" of ill-defined problems. According to Reitman, one may indeed lack well-specified algorithms for problems such as theorem proving, but the heuristic programs that, in these years, came to solve such problems were themselves "otherwise quite precisely defined". In addition, both the initial database from which the problem started and the test whereby one determined whether the problem had been solved, were well specified (Newell, 1969, pp. 366-367). However, where Reitman attempted to formulate "a positive characterization of problems by setting out the possible forms of uncertainty in the *specification of the problem*" (p. 367; *italic*s added), Newell tackles the question of ill-defined problems analysing the *activity of problem solving*, especially the *methods* proposed. He is conscious that, proceeding in this way, his approach "de-emphasizes some important aspects, such as the initial determination of an internal representation, its possible change, and the search for or construction of new methods (by other methods) in the course of problem solving". (Newell, 1969, p. 369) At the end of his paper, Newell comes back to these "important aspects" and extensively discusses a series of "difficulties" encountered by his position.

He starts his "Difficulties" section asserting that heuristic programming problem solving methods are "only a part of problem solving". We listed above some of "the many [other] parts of problem solving" that he enumerated (recognition, evaluation, representation, information acquisition, method identification, method construction, and executive construction and representation construction). Newell asks himself if "the aspects of problem solving that permit a problem solver to deal with ill-structured problems reside in one (or more) of these parts, rather than in the methods" (p. 407).

With respect to "method identification", e.g., he mentions that "much of the structuring of a problem takes place in creating the identification" (Newell, 1969, p. 409). For ill-defined problems, the difficulty of this creation resides in the identification from an unformalised environment to the problem statement of a precise, particular, formalised method.

Information acquisition, another example, seems not particularly specific to ill-defined problem solving. For well-defined problems, however, the search for information is guided by a highly specific goal; whereas, in the case of ill-defined problems, information is to be acquired for ill- or underspecified use: it is "to be used at some later time in unforeseen ways" (p. 410). Information acquisition could, therefore, play "a central role in handling ill-structured problems" (p. 410).

The major "difficulty" of his position is perhaps, according to Newell, that it does not "come to grips directly with the nature of vague information. Typically, an ill-structured problem is full of vague information. This might almost be taken as a definition of such a problem, except that the term vague is itself vague". (Newell, 1969, p. 411) Newell assumes that "the notion of vague information is at the core of the feeling that ill-structured problems are essentially different from well-structured ones" (Newell, 1969, p. 412). Of course, a problem solver has a definite problem statement, but "all the vagueness exists in the indefinite set of problems that can be identified with the problem statement.… When a human problem solver has a problem he calls ill-structured, he does not seem to have definite expressions which refer to his vague information. Rather he has nothing definite at all. As an external observer we might form a definite expression describing the range (or probability distribution) of information that the subject has, but this 'meta' expression is not what the subject has that is this information". (Newell, 1969, pp. 411-412)

Simon has also formulated some interesting observations with respect to the role of knowledge relative to the ill structured character of a problem —even if these observation remain isolated in his further "strict" approach to the structured character of problems and their solving. He

notes, e.g., that "ingenuity, whatever it is" generally makes us "[move] out into the broader world of ISPs" (ill structured problems) (Simon, 1973/1984, p. 185). Using ingenuity, indeed, generally requires violating the well-structuredness condition that "any knowledge that the problem solver can acquire about the problem can be represented in one or more problem spaces" (Simon, 1973/1984, p. 183).
Simon also states that "there may be nothing other than the size of the knowledge base to distinguish ISPs from WSPs" (well structured problems) (1973/1984, p. 197). He does not seem to consider, however, that such a position makes problem structuredness into a relative problem characteristic.
Surprisingly, he advances that "any problem solving process will appear ill-structured if the problem solver is a machine that has access to a very large long-term memory (an effectively infinite memory) of potentially relevant information, and/or access to a very large external memory that provides information about the actual real-world consequences of problem-solving actions". (Simon, 1973/1984, p. 197) I would, indeed, invert this relation —and expect, given Simon's previous remark about the role of knowledge, that he would also have advocated this inverse relation: a task will rarely constitute a problem —even less an "ill-defined" problem— for a person who has access to a very large long-term memory (an effectively infinite memory) of potentially relevant information, and/or access to a very large external memory that provides information about the actual real-world consequences of problem-solving actions.
Newell (1969)'s approach to this question may establish a bridge between Simon's position and mine. In the same way as Simon, Newell defends that humans do not have strong, i.e. domain specific, "methods of unknown nature for dealing with ill-structured problems" (1969, p. 407). Humans use the same range of heuristic, and other weak methods as they do for solving rather well defined problems. As presented above, Newell (1969), however, identifies that methods are "only a part of problem solving" and that there are "other parts" (presented above). Two of these other parts are "representation" and "representation construction" (1969, p. 407). Newell advances "the possibility that only special representations can hold ill-structured problems" and that to handle such problems is "to be able to work in such a representation". He has the "suspicion" that "changes of representation [at a certain level]… might constitute a substantial part of problem solving". (p. 408)

Thus, "abandoning" Simon as our (only) reference for the SIP approach to design, and referring (also, or instead) to Newell, allows nuancing our view of the SIP position concerning the ill definedness of design —and possibly other aspects of design.
NOTE. Both Newell (1969) and (Simon, 1973/1984) discuss the question of ill-structured problems in an Artificial Intelligence context. Simon, however, also extends his discussion to human problem solving.

4.2.2 □nderestimating the role of problem representation building

Simon has a rather inconclusive position with regard to the role of representation in design. As noted, Simon underlines the importance of representation in indeed several texts. Simon, however, nearly always advances his assertions concerning such importance in a final subsection, or he presents them as a detail that, even if it is said to be important, "deserves further examination", and is not central to a SIP model of problem solving.
The general reference model for problem solving, presenting its central components was presented in 1972 (Newell & Simon, 1972), whereas "The sciences of the artificial" had already been published in 1969. The 1996, third, edition is presented as a "revised" version in that it contains "new references" that record "the important advances that have been made since 1981 in cognitive psychology… and the science of design" (p. ix). Indeed, it presents numerous new references, but mainly in footnotes, whereas the core of the text has not been amended.
On several occasions, Simon starts with a very strict position concerning an issue we consider "critical", nuancing it little by little. With respect to representations, "finding" them or creating

them, Simon starts writing that "every problem solving effort must begin with creating a representation for the problem —a problem space in which the search for the solution can take place". He then continues: "Of course, for most of the problems we encounter in our daily personal or professional lives, we simply retrieve from memory a representation that we have already stored and used on previous occasions". Sometimes, he admits, we have to adapt an existing representation. "Occasionally", he declares, a new representation, a new problem space, has to be discovered. "More often", one is midway between simply adapting a known representation and inventing a new representational system.

When he suggests how a new representation comes to life, he states that "focus of attention is the key to success —focusing on the particular features of the situation that are relevant to the problem". (Simon, 1999, pp. 108-109) This focus-of-attention idea constitutes a first step toward building "a theory of representation change" as Simon calls it —we would rather speak of "a theory of representation construction". Simon also speaks of "discovering new representations" as if they have always been there, simply waiting to be found. He admits that the processes involved in this activity of discovery are "a major missing link in our theories of thinking" and notes that they are "currently a major area of research in cognitive psychology and artificial intelligence" (Simon, 1999, p. 109). To this respect, he refers to his study with Kaplan (1990), which indeed proposes some elements for such a theory.

It is again the chapter on social planning that leads Simon to expand the "Representation of design problems" topic (Topic 7 of his design curriculum). In this chapter, he also introduces at least six new topics, from "Bounded rationality" to "Designing without final goals" (Simon, 1999, p. 166). Simon notes that "the design tools relevant to these additional topics are in general less formal than those… described in the previous chapter" on engineering design (Simon, 1999, p. 166). The domain —or type— of design seems thus to play an important role in the types of cognitive processes and representations that are used —more or less— in the design problem solving. We will discuss this point in our final Conclusion.

As the authors mention explicitly, Newell and Simon's SIP theory of human problem solving does not deal with *changes* in problem representation (1972, pp. 90-91). In their 1972 vision, problem solving is search *in* a problem space constructed based on the problem specifications received at the start. The authors focus on the way in which, given a particular problem space, i.e., a particular problem representation, a person tries to solve that problem. Among the problem solvers observed by Newell and Simon, only one paid specific attention to *choosing* a problem space, i.e. a problem representation.

A metaphor used to characterise typical solution processes adopted in order to solve "typical" classical problems is "searching through the set of possibilities". The corresponding metaphor that Greeno and Simon (1988) propose for design problems is "narrowing down the set of possibilities" defined by a problem space. This is indeed the approach that Simon (1999) seems to adopt to design. At the beginning of the "Science of design" chapter, he asserts that design requires making a rational choice among a set of given, "fixed alternatives", "computing the optimum" (Simon, 1999, p. 114-119). Proceeding in this way is probably a "good", if not "optimal" approach. "Real" designers, however, i.e., professionals working on "real", actual design projects, often, if not generally, do not consider "the set of possibilities". Identifying *the* set of possibilities would indeed be a —too— heavy task (see "satisficing").

In a later section of his text, Simon indeed observes that his position is not realistic: "in the real world", design alternatives are "not "given in any constructive sense, but must be synthesised". They are even not "given" "in the quixotic sense that [the set of alternatives] is 'given' for the travelling-salesman problem" or "given" as the result of using a "formal but impracticable" algorithm (see also Newell, 1969's position, presented above, concerning the state of algorithms in the fifties of the xxth century). Therefore, designers, rather than starting with a set of given alternatives amongst which they have to choose, are concerned by "*finding* alternatives" (italics added), i.e. by synthesising such alternatives. "Once [they] have found a candidate [they] can ask: 'Does the alternative satisfy all the design criteria?'" (Simon, 1999, pp. 119-121)

"But how about the process of *searching* for candidates?" (Simon, 1999, p. 121) Simon proposes means-ends analysis, i.e. the general strategy proposed in 1972 as the strategy adopted in the classical laboratory problems (Newell & Simon, 1972). The plausibility of using this strategy in design will be discussed in the section "Overestimating the role of means-ends analysis".
Then, finally, near the end of his chapter, in a subsection entitled "Problem solving as change in representation", Simon writes that "a deeper understanding of how representations are created and how they contribute to the solution of problems will become an essential component in the future theory of design". "Alternative representations for design problems" then becomes the final subject in Simon's program on the theory of design (Simon, 1999, pp. 132-134).
In 1972, Newell and Simon (1972) provided some preliminary ideas concerning "problem redefinition", based on their analysis of De Groot's observations of chess masters. "Redefinition may come about (1) because a feature of the situation is noticed that has been overlooked earlier, (2) because initial expectations about the value of a move have been disappointed, or (3) because exploration of a move relates it to outcomes and goals different from those that suggested it. In these situations the dynamic analysis of moves appears more as an information-gathering process than simply as forward search through a branching space" (Newell & Simon, 1972, p. 762).
In the research conducted by Simon and colleagues on insight problems and scientific discovery, one may find some elements for this "future" theory of design in which an important role is attributed to representations, their creation and modification. This research is described as representing "a major extension of the standard information processing theory of problem solving" (Kaplan & Simon, 1990, p. 376). It brings, according to its authors, the "good news" that "the same processes that are ordinarily used to search *within* problem space can be used to search *for* a problem space (problem representation)" (Kaplan & Simon, 1990, p. 376). "But since previous evidence shows that subjects do not often switch their representations, [the authors] must also explain how the search for a new representation is initiated and under what conditions it has chances of success". (Kaplan & Simon, 1990, p. 376)

4.2.3 □nderestimating the role of "nondeterministic" "leaps"

Simon does not deny that processes such as intuition may play a role in expert activities. Intuition, however, is a "phenomenon which can be explained rather simply: most intuitive leaps are acts of recognition" (Simon, 1999, p. 89). So, rather than on search, intuition is based on recognition, which often takes the form of pattern-recognition in tasks in which perception may play an important role, such as in chess. In his section on intuition, indeed, Simon (1999) is concerned with chess grandmasters, whose "intuitive leaps" occur between the coding of physical chessboard position features and the memory representations of these positions as familiar configurations —i.e., between elements already associated in memory, i.e. separated by one "leap" (link). Furthermore, these intuitive leaps occur within a "closed" —even if vast— domain whose number of components is limited —even if large.
Elsewhere, Simon has advanced that the mechanism of production systems may explain intuitive thinking and insight: an experienced person in a domain is "simply" reminded of the solution to a problem. This solution "was retrieved by an action of recognition, the [problem] constituting a cue that evoked one or more appropriate productions" (1992a, p. 133). And Simon to quote Pasteur: "As Pasteur once put it, 'Chance favors the prepared mind.'" (ibid.)
"Interesting" design ideas often depend on "leaps" *between* domains —which immediately increases not only the number of possibly relevant candidates, but also diminishes the chance of associative links already existing between problem feature representations and candidate solution representations in memory.

□*wo examples of "leaps"*. In our Composite-Structure design study, we observed designers performing "leaps" between, e.g., the problem of designing "unfurling principles" for future antennas, and (mental representations of) "umbrellas" and other "folding" objects, such as "folding

photo screens," "folding butterfly net," and "folding sun hats". It seems implausible that the corresponding "intuition" relied on pre-existing associations (cf. Johnson-Laird's "pre-existing rules between source and target", presented below).
Neither do we suppose this to be the case for leaps between the problem of designing a support system for such an unfurling antenna, and "curtain rails", "train rails" and "railway catenaries". These were analogical leaps between domains whose elements did bear no surface similarity.

It seems to us that, in order to explain such analogical leaps, more complex structures or mechanisms are required —even if not instead of, but rather in addition to, recognition.
There have not yet been many serious proposals for such a type of reasoning by analogy. One exception is the analysis put forward by Johnson-Laird (1989) for what he calls "profound analogies". This author notices that there are, of course, forms of analogy "that can be retrieved by tractable procedures". Johnson-Laird argues, however, that "the processes underlying the discovery of profound analogies.… cannot be guaranteed by any computationally tractable algorithm" (p. 313). Profound analogies involve "genuine human creativity", which Johnson-Laird (1989) claims is "nondeterministic".
"The creation of a profound analogy is unlikely to depend on preexisting rules that establish mappings between the source and target domains. The innovation indeed depends on the invention of such rules.… No algorithm for searching the correct mapping can run in a realistic time beyond a certain number of links". (p. 327)[35] The critical step in creating a profound analogue consists in discovering the relevant source domain. "The more constraints that individuals can bring to the task —the more they know about potentially relevant domains— the more likely they are to find the illuminating source". (pp. 329-330)
NOTE: Reasoning on the ill-defined problems involved in the composition of a fugue, Reitman (1964), however, notices that "though they would generally be considered complex they include few constraints as given" (p. 296).
Very modestly, Johnson-Laird (1989) concludes that previous theorists have emphasised before many points made in his paper. "What, perhaps, has not been noted before are the computational consequences of the exercise of creativity in the discovery of original analogies". (p. 330)
In order to defend the straightforward, elementary nature of ill-defined problem solving, Simon presented the basically serial character of the problem solving system, and its limited capacity confining it to work on only a few input elements and to produce only a small number of symbol structures as output[36]. The same arguments can be advanced, in our opinion, in favour of Johnson-Laird's position. It seems to us that the number of associative links that are to be traversed between source and target structures, each in remote domains, in order to lead to what Johnson-Laird (1989) calls "profound analogies" cannot be traversed with some acceptable degree of probability and/or in some acceptable time scale. They cannot, neither by forward (prospective)

[35] In a description of the EPAM program, built to simulate human rote verbal learning, Simon (1995) presents a tree-like discrimination net as the core of the system. He writes that "if the net has a branching factor of 4, then recognition of a net discriminating among a million stimuli could be achieved by performing about ten tests (4^{10} = 1,048,576). The EPAM model, its parameters calibrated from data in verbal learning experiments, can accomplish such a recognition in a tenth to a fifth of a second". (p. 942) This seems impressive, but we don't know how to estimate the distance (i.e., number of transitions) between representations corresponding to, e.g., an antenna "unfurling principle" and an "umbrella". In the context of his analysis of ill-structured problems, however, Simon states: "There is no way in which a large amount of information can be brought to bear upon these processes locally —that is, over a short period of processing. If a large long-term memory is associated with a serial processor of this kind, then most of the contents of long-term memory will be irrelevant during any brief interval of processing". (Simon, 1973/1984, p. 192)

[36] See Note 34.

search (generation and test), nor by backward (regressive, retrospective) search (means-ends analysis).
This point is related to the "mechanization" premise of the SIP model, questioned by Mayer (1989, pp. 54-55) in his paper on "human nonadversary problem solving" (in which he presents a series of arguments calling into question several implicit premises of the SIP model). Mayer notices that many problems, especially ill-defined ones such as design problems, require strategies that are "much less algorithmic [thus less "mechanizable"] and more intuitive than means-ends analysis" (p. 55). It may also illustrate the SIP premise that Mayer qualifies as "concretization". In the SIP approach, each action results in "movement from one concrete state to another. However,… thinking sometimes occurs at a general or a functional level rather than at the level of specific problem states". (ibid.) Indeed, in order to come up with "new" ideas, people may need to break completely from their current line of thinking.
Simon, however, does not consider the need for such "nondeterministic" leaps —or perhaps their possibility. He admits that design takes place in a context in which "all potentially relevant information" is not fixed, not even present, right from the start[37] —as is required for a problem to be considered well defined. Simon yet reasons as if all problems —be they design or other types of problems— may be solved in one step: either between a process and a subprocess that this process calls via a subroutine structure, or between two symbol structures, the first one evoking the second one via this mechanism that "recognizes when certain information has become relevant" (Simon, 1973/1984, p. 193).
As noticed above, for Simon, "insightful" problem solving does not call for specific processes —"creative" processes— that are different from those observed in all kinds of problem-solving settings. To this respect, Simon resembles his behaviourist ancestors, with whom his theories were in strong opposition. Behaviourist theorists also advanced that the same processes are involved in solving routine and nonroutine problems. It were the Gestalt theorists who judged that "nonroutine (or creative) problem solving involves qualitatively different thinking than routine problem solving" (Mayer, 1989, pp. 51-52).

4.2.4 Overestimating the role of problem decomposition

Another example of Simon adopting an approach to problem solving that cannot account for design —at least in usual industrial design projects— is the role attributed to the top-down problem-solving method of decomposition.
Referring back to Christopher Alexander (1964)'s famous hierarchical decomposition method (see Alexander, 1963/1984), Simon posits problem decomposition as a powerful problem-simplification method, particularly useful in solving ill-defined problems.
NOTE. To this respect, Simon's conception of human design resembles much the "transformation approach to design" frequently adopted in Artificial Intelligence (Balzer, 1981; Barstow, 1984). Brown and Chandrasekaran (1989, pp. 21-22) critique this approach on at least four points.

- In some domains, the constraints as stated may not be factorisable as anticipated, and there may be significant interactions between the designs that are chosen to meet parts of the constraints.
- There is no guarantee that the design process can always correspond to incremental choices. Large subsystems may be designed first and only then can design proceed within subsystems. Thus the actual design process in such a domain may not correspond to navigation in this transformation problem space.
- Knowledge may be directly available which cuts a swath across the space, so that several constraints together are realized by a precompiled design that is recognized as applicable.

[37] Simon even writes that "there is no need for this initial definition" (Simon, 1973/1984, p. 193).

- Finally, in many domains, the problem is reformulated by a decomposition so that a number of disjoint local spaces, each corresponding to a subproblem, are created.

Brown and Chandrasekaran (1989) consider that one of the characteristics of "open-ended creative design" is the absence of a "storehouse" of effective decompositions and of design plans for subproblems. In case problem decomposition knowledge is available, most of the effort is in searching for potentially useful decompositions. The authors suspect that very little design activity is in this class of "extremely innovative behavior" "leading to major inventions or completely new products" (p. 33).
In "relatively routine design", one knows "effective problem decompositions", "compiled design plans for the component problems", and "actions to deal with failure of design solutions". "There is very little complex auxiliary problem-solving needed. It is not trivial, however, as plan selection is necessary and complex backtracking can still take place. The design task is still too complex for simple algorithmic solutions or table look up.… [It still requires] knowledge-based problem-solving". (Brown & Chandrasekaran, 1989, p. 34)

Both empirical studies and theoretical analyses show that, even if decomposition may indeed be a strong and useful method, its application often is not smooth at all (see also Brown and Chandrasekaran's observations).
Decomposition is not a simple procedure that designers may execute in a straightforward manner. First, the multiple interdependencies among the subproblems resulting from a first decomposition (one of complex design problems' characteristics in our approach) "are likely to be neglected or underemphasized", as Simon notes himself. "Such unwanted side effects accompany all design processes that are as complex as the architectural ones we are considering". (Simon, 1973/1984, p. 191)
Second, one and the same design component often can be decomposed in different ways (Reitman, 1964, p. 296).
A third complication concerns the application of "transformational formulas". Decomposition can indeed be implemented based on a repertoire of decompositional rules and methods possessed by professionals experienced in the domain. In this line, Simon argues that design acquires structure through the application of "formulas" such as: "house" transforms to "general floor plan plus structure", then "structure" to "support plus roofing plus sheathing plus utilities", etc. (Simon, 1973/1984, p. 190). Reitman (1964), to whom Simon refers for his "transformational formulas" approach, has imported the idea as an analogy from structural linguistics. He notes, however, that although it is "a useful analogy, it must not be carried too far". There is no fixed limit on the number of sources of transformational formulas that a designer may consult. Besides, new formulas may be developed. Furthermore, based on unforeseeable decisions and information sources, decomposition may follow another than the specified order.
In a study of social sciences problem solving (particularly in the political science domain), Voss, Greene, Post, and Penner (1983) observed two modes of subproblem generation. Subproblems resulted either from deliberate decomposition, or they were "encountered" when problem solvers explored the implications of proposed solutions. Experts primarily used this "encountering" mode. Novices typically proceeded by problem decomposition. The authors' comparative analysis of four groups of problem solvers[38] leads them to conclude that the use of the problem decomposition strategy may be analysed as the falling back upon more general problem solving strategies when one is confronted with a problem outside one's specialisation —i.e. when one is, or becomes again, a novice.

[38] These were (i) experts in the domain, (ii) novices in the domain, (iii) graduate students that formed a "transition" between these experts and novices, and (iv) "nonexpert experts" (i.e., advanced graduate students and faculty members in political science, but whose field was not the particular problem domain, i.e., the Soviet Union).

NOTE: This conclusion concurs with our observations concerning the "counterintuitive data from empirical studies" concerning the differential use of opportunistic strategies by novices and experts (Visser, 1991c).
A characteristic of decomposition that is not necessarily a drawback of the method is its generally leading to standard, routine solutions. Indeed, decompositions are based on *a priori* categories, leading to the attribution of "default" values to the different components (see also Carroll, 2000, p. 27). This may be what a designer searches for, but will not be systematically the case. This strengthens Voss and all (1983)'s interpretation.

4.2.5 Overestimating the role of means□ends analysis

The main problem solving method proposed by Newell and Simon (1972) as identified in their studies of their classical laboratory problems, is means-ends analysis. As Simon (1999) defends the "nothing special" view with respect to ill-defined problems —no new concepts or techniques are needed in order to solve these problems— this method is apparently also supposed to be used in order to solve design problems. In *Sciences of the artificial*, Simon indeed explicitly proposes means-ends analysis in response to the question: "But how about the process of *searching* for candidates?" (Simon, 1999, p. 121)
In our view, however, this *heuristic* will generally be inappropriate for solving design problems. Indeed, in order to use it, a designer must be "able to represent differences between the desired and the present". "The desired" is "the" problem's goal state; "the present" is the problem's current state (an intermediary problem state). If differences are found, a mechanism equivalent to GPS is brought into play. GPS is the computer problem-solving program designed to "model some of the main features of human problem solving". It uses a "table of connections, which associates with each kind of detectable difference those actions that are relevant to reducing the difference". When a difference is detected, GPS "searches selectively through a (possibly large) environment in order to discover and assemble sequences of actions that will lead from a given situation to a desired situation" (Simon, 1999, pp. 121-123).
For design problems, however, it will generally be difficult to establish the differences between these two states —that is, if ever "the" goal state has already been specified. Even if both states are specified, they often are at different levels: during design-problems solving, the current state of the problem may have been specified at a "concrete" level, whereas its goal state will generally, at best, be specified at an abstract level.
Even if differences can indeed be detected between the two states, complications may arise with complex problems —as design problems often turn out to be. The integration of "sequences of actions" or other solution-components, into one global solution comes up against their non-additivity, their non-independence, and their interaction. Moreover, "GPS operates… on formally presented problems, not on an external real world" (Simon, 1973/1984, p. 184).
Means-end analysis will thus probably rarely be an appropriate heuristic for handling design problems.

4.2.6 Overestimating the role of search

As noticed above, search in and/or through problem spaces is central in the SIP approach to problem solving. An information-processing definition of "problem" illustrating this role, is the following: "a problem exists when an information-processing system has a goal condition that cannot be satisfied without a search process" (presented by Gilhooly, 1989, p. 2, as the information-processing rephrasing of Duncker's Gestalt definition of "problem solving" quoted above). "The" solution to a problem is "found" by a "search" that constitutes an "odyssey through the problem space" (Simon, 1978, p. 276).
Applied to the problems we are interested in here, this gives: "Design problems can be understood as problems of search in a space that contains many possible arrangements of the problem materials, only one or a few of which satisfy the problem criterion" (Greeno & Simon, 1988).

There are authors who distinguish "design problems" from "search problems". For Logan and Smithers (1993), e.g., in a design problem "either the objective to be achieved or the means of achieving it (or both) are initially only poorly defined. Design in this sense is explicitly not a *search* process in which the task is essentially one of selection or optimisation over a completely defined space". Design involves solution *generation*.
Notice that Logan and Smithers adopt the same position as we do, when they consider that "design" is "creative design". The authors prefer to use explicitly the term "routine design" for other tasks that designers also handle of course and that may, e.g., be solved completely by search. Many sub-problems of a global design project will indeed be solved by such "routine design".
Logan and Smithers propose to model design as knowledge-based exploration (see also Navinchandra, 1991, who considers exploration an important aspect of what he qualifies as "innovative" design). This design problem exploration goes together with problem redefinition. Design problems' initial requirements are incomplete and inconsistent. In order to identify these omissions and inconsistencies, knowledge of the space of possible designs (SPD) is required. "The initial requirement description cannot therefore serve as a specification for a search problem — since if it is incomplete and inconsistent it cannot define a goal-state in a SPD". (Logan & Smithers, 1993)

Problem solving is not only search. In terms of the SIP framework, design problems are "search" problems, as are all other types of problems. In their concluding Theory section of *Human problem solving*, Newell and Simon indeed "postulate that problem solving takes place by search in a problem space. This principle is a major invariant of problem solving behavior that holds across tasks and subjects". (1972, p. 809) In his conclusion of *The sciences of the artificial* chapter on "The psychology of thinking"[39], Simon states that "the theory of design is that general theory of search", i.e. "the general theory of search through large combinatorial spaces on the outer side [of the human brain, that is] the side of the task environment" (Simon, 1999, p. 83).
Newell and Simon, however, also notice in their concluding Theory of *Human problem solving* chapter that "[the statement that problem solving takes place by search in a problem space] does not mean that all behavior relevant to problem solving is search in a problem space". (1972, p. 809) For example, "defining the situation", which may lead to "problem redefinition", "contrasts sharply with search activity" and thus, "is an important type of information-processing to understand" (p. 761).
In their 1972 book, however, Newell and Simon do not develop this topic. It was going to be the object, e.g., of the Kaplan and Simon (1990) study that we discussed above in our consideration of Simon's underestimating the role of problem representation building.

We want to conclude this section by a distinction between "exploration" and "search" (in problem spaces). We consider that "search" is concerned with identifying a particular solution, whereas "exploration" concerns finding something that can only be identified as interesting or valuable once it has been found. Exploration involves the generation of design solutions before exploring these different alternatives and choosing one (see also Navinchandra, 1991).

4.2.7 Simon's "more nuanced" positions in more recent texts

Simon's position with respect to design, and to related essential matters discussed in the present text —construction of representations, the predominance of search in problem solving— is not consistent. Understandably, more "nuanced" positions appear in more recent texts, mostly on scientific discovery and creative thinking (Cagan et al., 2001; Kaplan & Simon, 1990; Klahr & Simon, 2001; Kulkarni & Simon, 1988; Simon, 1992b, 1992c, 1995, 2001a, 2001b). There is, however, among his two or three rare publications explicitly dealing with design, also an *early*

[39] This chapter from 1969 was maintained as such in the 1996 Third revised edition.

paper in which Simon presents a position much closer to the proposal formulated in this text, i.e. to consider design problems as ill-defined with respect to all three problem components, with the goal state being relatively least ill-defined (see below). The paper —*Problem forming, problem finding, and problem solving in design*— published in 1995, was based on a conference presented in 1987, that is several years after Simon's famous *The structure of ill-structured problems* (1973). In this 1987/1995 paper, Simon, e.g., states: "We cannot really regard the goals of design as given any more than we regard [the initial states] as given. A design process begins with some criteria and some possibilities (or primitives out of which alternatives can be constructed). As the process goes forward, new criteria and new possibilities are continually being evoked from the sources we have identified". (1987/1995, p. 253)
In Simon's 1973 paper on ill-structured problems, which was already largely quoted and discussed above, there is also an interesting assertion concerning the illusory character of problem structuredness. "Definiteness of problem structure is largely an illusion that arises when we systematically confound the idealized problem that is presented to an idealized (and unlimitedly powerful) problem solver with the actual problem that is to be attacked by a problem solver with limited (even if large) computational capacities". (Simon, 1973/1984, p. 186) This claim was, however, isolated in the further context of the paper. It did not lead Simon, e.g., to conclude that, for human problem solvers in any case, all "real" problems are ill structured .
Surprisingly, also —as already noticed above— Simon did not integrate this position, neither in the second, 1981, nor in the third, 1996, edition of *The sciences of the artificial* (first edition 1969), his most prominent publication on design, which constitutes *the* reference for his approach in this domain.

Scientific discovery and "inventive" design. Most "more nuanced" positions thus appear in texts concerning scientific discovery and creative thinking. In a paper presenting the state of the art in 2001 concerning psychologists' —and other researchers'— discoveries about the process of scientific discovery, Klahr and Simon (2001, p. 76) affirm that "initial [problem] state, goal state, operators, and constraints can each be more or less well-defined. For example, one could have a well-defined initial state and an ill-defined goal state and set of operators (e.g., make "something pretty" with these material and tools), or an ill-defined initial state and a well-defined final state (e.g., find a vaccine against HIV). But well-definedness depends on the familiarity of the problem-space elements, and this, in turn, depends on an interaction between the problem and the problem solver".
Another recent, particularly interesting paper compares scientific discovery and inventive engineering design, with respect to their cognitive and computational similarities (Cagan et al., 2001). In the same way as with respect to the general mechanisms that may be used in order to solve design problems and ill-structured problem, the authors defend the "nothing special" with respect to insight and scientific discovery problems. "Perhaps the most important feature of the theory [of discovery] is that it is built around the same two processes that have proved central to the general theory of expert human problem solving: the process of recognizing familiar patterns in the situations that are presented, and the selective (heuristic) search of a problem space. Basic to these are the processes of formulating a problem space and of representing available information about the problem, as well as new information acquired through observation and experimentation, in that problem space". (Cagan et al., 2001, p. 450)
In these recent papers, one may find other statements concerning the important role of representation in problem solving. "Representational changes" play a role in scientific discovery in that they enable scientists "to replace [an] entrenched idea" with new ideas (see also Kaplan & Simon, 1990; Klahr & Simon, 2001, p. 77)
In a paper on the importance of "curiosity" for "discovery" (see also Simon, 2001a; Simon, 2001b), Simon posits that —at least in certain cases— "creative" ideas and/or hypotheses, come from "noticing some phenomena" and "recognizing some things about them" (Simon, 2001b, p. 10). Even if Simon does not insist on the role of knowledge, he mentions that something we

would call "prior knowledge" conditions the occurrence of these processes: "we mainly notice things that are unusual or surprising in their current surroundings", that is, "when we have expectations that are violated" (Simon, 2001b, p. 10). Different types of data sources "all provide evidence that the scientist's reaction to phenomena… that are surprising can lead to generating and testing new theories". (Klahr & Simon, 2001, p. 78)
"Scientific discovery is a type of problem solving using both weak methods that are applicable in all disciplines and strong methods that are mainly domain-specific.… Recognition processes, evoked by familiar patterns in phenomena, access knowledge and strong methods in memory, linking the weak methods to the domain-specific mechanisms". (Klahr & Simon, 2001, p. 78)
The role that Simon attributes to *surprise* in discovery may be related to the description that Schön and Wiggins (1992, Spring) provide of "seeing-moving-seeing" "snippets" as possibly leading to the "discovery of unintended consequences" of design actions.
This observation may, in its turn, be related to the role that Schön (e.g. 1992, March, p. 7, quoted above) assigns to recognition in such discoveries and that is central in the explanations in Simon's later work.
In their comparison between design and discovery, Cagan, Kotovsky, and Simon (2001) advance that "[highly creative] design activities are often labeled *invention*". They explain how the "seemingly disparate activities" of discovery and invention are surprisingly similar (Cagan et al., 2001, p. 442). The authors' "major conclusion" of this comparison is that "at a deep level, the cognitive and computational processes that accomplish [design and discovery] are virtually identical" (Cagan et al., 2001, p. 463). Their "real similarity" is made up by "the underlying cognitive activities based on problem solving, pattern recognition, analogical reasoning, and other cognitive knowledge retrieval mechanisms" (Cagan et al., 2001, pp. 452-453).
"The fundamental difference" between the two is "the *goal* of the process: scientific explanation versus creation of a new artifact.… Design starts with a desired function and tries to synthesize a device that produces that function. Science starts with an existing function and tries to synthesize a mechanism that can plausibly accomplish or account for that function". (ibid.) However, "their cognitive models of search, once the problems are defined, are the same". (Cagan et al., 2001, p. 455)
An implementation of these ideas can be found in A-Design, a program proposed by Campbell, Cagan, and Kotovsky that automates the invention of electromechanical artifacts (1999; 2000). "At a high level the A-Design program is a model of (human) group activity". (Cagan et al., 2001, p. 459) The observation that in human creative design, "random directions are chosen and pursued if seemingly beneficial or else reversed if seemingly inferior" motivates, "on a high-level analogy", the search strategy of "simulated annealing" in A-Design and similar programs (Cagan et al., 2001, p. 461).
NOTE. Several authors have discussed the relation between design and science and between design and discovery. In an analysis of the structure of design processes, Dasgupta (1989), e.g., considers design problem solving is a special instance of scientific discovery (Design-as-Scientific-Discovery, DSD).

4.2. Conclusion concerning the SIP approach

The six aspects of design misrepresented in the classical symbolic information-processing approach and discussed in this section, constitute, depending on one's optimism and/or one's confidence in the SIP approach and its adaptability, either a series of nuances with respect to this approach that might guide its modification, or such severe critiques that one judges that the approach is to be abandoned.
We have discussed, as were they two in parallel existing, "types" of positions adopted by Simon. First, we saw his rather "strict" SIP position concerning design as regular problem solving that deserves, and needs, "nothing special". Afterwards, this section introduced a much more nuanced position, defended, on the one hand, in papers concerning discovery and creative thinking, and on the other hand, in the paper with Cagan and Kotovsky concerning inventive engi-

neering design. We may relate these different positions defended in parallel by Simon, to what we have noticed above concerning Simon's divergent positions with respect to social planning compared to design thinking in general. We indeed observed that Simon considered "standard" engineering and architectural design as a "simple" problem solving activity comparable to that required by the classical laboratory problems. Simon presented design of social plans, however, as an activity in which, e.g., representation problems play an important role.
One might explain these differences if one may suppose that for Simon
- "standard" design was *routine* engineering and architectural design;
- social planning was *radically different* from such "standard" design; and
- inventive engineering design was *a special form* of design.

The creative character of discovery and invention would "normally" not apply to design. However, in the above-presented paper on inventive engineering design, Cagan, Kotovsky, and Simon (2001) claim that "invention is not essentially different from other types of design activities" (p. 451). The "nothing special" position traverses all Simon's work. This may be attributable to Simon's aspiration of one global theory that covers all problem solving.

4.3 Discussion of the situativity approach

Given that the situativity approach to design has been presented based on Schön's approach to design (with some references to Bucciarelli and Gero), Schön will be central in this discussion of the situativity approach (but see hereunder our paragraph "Confronting SIP and SIT approaches"). This discussion will be much shorter than that of the SIP approach, mainly because the situativity approach provides us with a global viewpoint on design (and other activities) and therefore invites a global discussion, whereas the SIP approach has made many precise proposals that have led to corroboration or objections in the empirical studies that constitute our working basis.
Each research community chooses one or more focuses of interest for their research —at the price of simplifying or even neglecting other topics. Our global critique at the situativity approach is that
- data has an anecdotal flavour —provoking questions such as "Interesting, but so what?";
- the methods used for data collection and analysis, and consecutive modelling, offer no tools for deriving higher-level descriptions from sets of observations (see also Nardi, 1996a, p. 83); and
- the analyses conducted from this viewpoint often present rich, detailed descriptions of activities, but also depict these activities as so unique that the results are unlikely to be replicable, and the conclusions (if ever there are some general conclusions) generalisable across situations. As remarked by Nardi (1996a, pp. 93-94) in her comparison of activity theory, situativity theory, and distributed cognition, SIT models "do not account very well for observed regularities and durable, stable phenomena that span individual situations".

These remarks apply entirely to Schöns approach to design. "Schön, himself, once described reflective practitioner research as 'non rigorous inquiry' (Schön, 1987, p. 3)", as spotted by McMahon (1988) in a paper discussing the similarities between theoretical conceptions of reflective practice and action research. Schön indeed does not proceed to an analysis of the structure of "reflective practice" —and neither does Bucciarelli. Adopting "ethnographic perspectives", these authors propose detailed, narrative presentations of design projects. Indeed, the authors show many, detailed aspects of the design projects they have been observing —and, for Bucciarelli, in which the author has been participating— giving a vivacious flavour of the richness of all activities that work on a design project may embrace.
In his review of *The reflective practitioner* referred to above, Ian Alexander[40] regrets that Schön, who provides interesting analyses of specific work situations, "which he discusses in

[40] Not to be confounded with the famous urban-design researcher and theorist Christopher Alexander, also referred to in this text.

detail to bring out how reflection fits into the professionals' use of knowledge", doesn't "go on to elaborate the structure, rules, and techniques needed to conduct the process efficiently.… He disappointingly stops short of doing this". (2001, Retrieved December 8, 2003, from http://i.f.alexander.users.btopenworld.com/reviews/schon.htm) Ian Alexander quotes Schön himself: "Reflection-in-action is a kind of experimenting.… In what sense, if any, is there rigor in on-the-spot experiment?. … Questions such as these point to a further elaboration of reflection-in-action as an epistemology of practice. One might try to answer them by appeal to a structure of inquiry, but I do not know what such a structure might be or how it might be discovered". Ian Alexander notes: "And [Schön] goes on to say that he'll look for answers in the documented examples" (2001, Retrieved December 8, 2003, from http://i.f.alexander.users.btopenworld.com/reviews/schon.htm).

4.4 Confronting SIP and SIT approaches

In his *Introduction* to the *Cognitive Science* special issue, Norman states that, in his view, the "two traditions do not seem to be contradictory.… They do not conflict". "They emphasize different behaviors and different methods of study". (1993, p. 3). However, "the social structure of science is such that individual scientists will justify the claims for a new approach by emphasizing the flaws of the old, as well as the virtues and goodness of the new. Similarly, other scientists will justify the continuation of the traditional method by minimizing its current difficulties and by discounting the powers or even the novelty of the new". (Norman, 1993, p. 3) Vera and Simon (1993a), the two authors representing the SIP approach, argue, e.g., that "the new approach could easily be incorporated within the old" (Norman, 1993, p. 1). For Anderson, Reder and Simon (1996), Vera and Simon (1993a)'s review is "in support of the mutual compatibility of modern information processing theory and situated cognition" (p. 5).

NOTE: In a paper from 1978 (1978), Simon made a statement concerning the scope of the SIP approach that sounded as a more modest claim. He stated that "information-processing theories have made especially good progress in providing explanations for solving relatively well-structured, puzzle-like problems of the sorts that have been most commonly studied in the psychological laboratory". (Simon, 1978 □2150, p. 272)

We may understand that researchers, such as the SIT authors judge that, rather than the underlying symbol-based mechanisms, the conditions of sense giving merit more attention and more study. They may selectively focus on the social and/or interactional aspects of these conditions. For Anderson, Reder, and Simon (January/February 1997), "the situated movement has performed a valuable service in emphasizing the important contextual and social aspects of cognition. However, the situated position has not shown that it provides the right theoretical or experimental tools for understanding social cognition". (p. 20)

We completely agree with Norman that the SIP and SIT frameworks are not contradictory. We consider that, focusing on different aspects of design, adopting different approaches, they make different contributions to design theory, which, in our view, are complementary. Studies conducted by reference to the SIP paradigm pay more attention to people's *use* of knowledge and representations than to their *generation*, particularly to the role that is played in this *generation* by factors such as interaction with other people and/or interaction with the larger environment.

SIT authors —if it is possible to speak about all these diverse researchers as one community— pay more attention to people's *interactions* and the role on people's actions played by their *environment*, the *social and cultural setting*, and the *situations* in which people find themselves. Remains —in our view as a researcher in cognitive psychology and cognitive ergonomics— the issue of the cognitive processes and representations that may account for such (inter)action. We are interested by questions such as: "Why do different people act differently in 'identical' situations"? We are not satisfied by answers that refer to the "environment", to the "context", because this doesn't answer the question: "What makes that different people 'interpret' differently the 'same' environment, an 'identical' 'context'?"

Two of the proponents of the SIT approach, Greeno and Moore (1993), agree with their "opponents", Vera and Simon, "that 'breaking completely' from symbolic cognitive theories would be the wrong thing to do, but [they] believe that something like 'departing fundamentally' is required". (1993, p. 57) "Within the historical development of psychology", they view, "in the present situation, a prospect of completing a dialectical cycle, in which stimulus-response theory was a thesis, symbolic information-processing theory was its antithesis, and situativity theory will be their synthesis". (1993, p. 57). They "contend that symbolic processing theory presents another black box that contains the structure of interactive relations between cognitive agents and the physical systems and other people they interact with. Vera and Simon assert that 'The symbolic approach does not focus narrowly on what is in the head without concern for the relation between the intelligent system and its surround' (p. 12). Even so, this concern has not led to analyses of agent-setting interactions in anything like the detail that has been characteristic of analyses of hypothesized cognitive structures and procedures". (Greeno & Moore, 1993, pp. 57-58)

We agree with this conclusion by Greeno and Moore: the SIP approach's "concern for the relation between the intelligent system and its surround" has indeed not been significant in SIP studies. Different things interest SIP and SIT researchers!

In their reply to Clancey (1993), Vera and Simon (1993c) notice that "complex systems are best described in several levels, each level corresponding to a different time frame. In the case of human behavior, events in the range from a few milliseconds to a few tens of milliseconds are best described in a vocabulary of neurons and their actions [by neuropsychologists]; events in the range from a few hundreds of milliseconds to a few minutes are best described in the operation of a symbol system with a large memory, relatively limited input and output devices, and a modest capability of learning (within this time frame) [by information-processing psychologists]. Events in the range from hours upward are best described in terms of collections of interacting symbol systems that influence not only each others' behaviors but also each others' memory stores [by social psychologists or sociologists]". (Vera & Simon, 1993c, p. 130)

For Greeno (January/February 1997), "the cognitive perspective takes the theory of individual cognition as its basis and builds toward a broader theory by incrementally developing analyses of additional components that are considered as contexts. The situative perspective takes the theory of social and ecological interaction as its basis and builds toward a more comprehensive theory by developing increasingly detailed analyses of information structures in the contents of people's interactions. While [he believes] that the situative framework is more promising, the best strategy for the field is for both perspectives to be developed energetically". (p. 5)

Dorst (1997), in his Ph.D. thesis, undertakes a comparison between Simon's and Schön's approaches, which he qualifies as "the two fundamentally different paradigms that the field [of current design methodology] is based on" (p. 204). He analyses the two "paradigms", and then conducts an empirical study focusing on "one of the key issues of design-as-experienced" (referring to Schön's approach), that is, the "integration" activity in design. The two paradigms are evaluated in terms of their ability to describe this activity. One of Dorst's main conclusions is that they describe distinct parts of the total design activity. One paradigm is more appropriate for describing activities in one design phase, the other for describing activities in another phase. The descriptions by Simon's "rational problem solving" paradigm perform best in design's "information phase". Examined from the problem-solving perspective, a designer works "as if [design involves] objective interpretation". Schön's design perspective is most useful for describing the "conceptual design" phase. Analysed from this viewpoint, a designer looks at design "as if it [involves] subjective interpretation". This conclusion leads Dorst to judge that the choice of the appropriate research and/or modelling paradigm depends on three main factors: the research goals, the objects of study, and, most importantly, the kind of design activity that is to be studied (pp. 166-167).

In our view, even if SIP and SIT frameworks are not contradictory, but complementary, both neglect aspects of design. Simon, in his discussions of design, disregards the "rich" and specific

characteristics of the activity pointed out above (the specificity of ill-defined problems and their solving, the role of problem representation building, etc.); Schön leaves us with detailed descriptions of extremely "rich" situations, but doesn't make the reflective step considered as critical, and required —in our opinion— in order to get to a more general level that yields statements about design "in general".
We agree with authors such as Schön with respect to the importance of problem "setting" and "(re)framing", the "active" and "constructive" aspects of problem understanding, which leads to a continuously evolving problem representation. For situativity researchers, the situational context is the "dominant resource". SIP researchers do not pronounce themselves in such terms, but they would probably refer to methods as the essential "resources" (see Newell, 1969).
We agree with authors such as Simon concerning the importance of people's internal, that is, mental representations, which, even if they evolve under the influence of interaction with people's environment (human and artificial, social and cultural), provide them with a "basis" from which they act on this environment. Situativity researchers do not deny the existence of such an internal apparatus, but they do not posit is as a "basis" and they do not give it much attention.
For a person confronted with a task, all resources for action depend on both internal (knowledge, representation) and external (generally, artefactual) data. In our view, these two types of data are, however, not in a symmetric relation: their interaction depends on the person, i.e. on their knowledge!

□□□

The presentation of our own approach to design, which constitutes the second half of this section, will show the influence of, and congruence with, both "traditions" that it will (try to) integrate, but also complete.

4.5 Definitions of design and our approach

As noted, among the four characteristics of design proposed by Simon and presented above, we quite adhere to Simon's qualifying design as a particular type of cognitive activity in which satisficing plays a particular important role. The two other characterisations translate a view that we do not share. They represent design as an activity much more systematic than empirical studies of actual design projects have shown design to be.
Before proposing our own definition, we will present some other definitions and characterisations of design that have been proposed in the literature. After the presentation of our global definition of the design activity, we will elaborate upon this definition and characterise design from two viewpoints, the design activity and the type of problems concerned by this activity.
Depending on the author, design consists in producing specifications (or plans, descriptions, representations) for an artefact (or system, device, procedure) (Archer, 1965/1984; Brown & Chandrasekaran, 1989; De Vries, 1994; Hoc, 1988; Jeffries et al., 1981; Kitchenham & Carn, 1990; Schön, 1988; Whitefield, 1989). Applied to software design, e.g., this means that design leads to a plan allowing transforming the specifications into executable code (Jeffries et al., 1981; Kitchenham & Carn, 1990). According to most definitions, the result of the design activity has to meet certain requirements, possess certain characteristics, satisfy certain constraints, allow to attain certain goals, and/or fulfil certain needs (Archer, 1965/1984). Several authors, especially in Artificial Intelligence, consider design a constraint-satisfaction activity (see Darses, 1990, for a discussion of this viewpoint from a cognitive psychology viewpoint). All these definitions of design refer to the goal-directed aspect of the activity (fulfilling needs, accomplishing functions, satisfying constraints).
Several authors from the engineering design domain propose comparable definitions. The definition that we proposed in our Functional specification study —"Design consists in transforming a problem representation into another: it… starts with 'requirements' and produces 'specifications'" (Visser, 1990)— joins particularly well the "more general definition" that Hubka, in an overview paper on engineering design, has proposed on the basis of the "common features found in [several] definitions" proposed by authors in the domain of engineering design (1987, p.

124). Several engineering design authors also mention the "purposeful", "goal-oriented", "human needs fulfilling" aspect of the activity into their definition (e.g., Asimov, 1962; Hubka & Eder, 1987).

Our definition of design. Globally defined, design as an activity consists in specifying an artefact, given requirements that indicate one or more functions to be fulfilled and/or objectives to be satisfied by that artefact. The cognitive activity corresponding to this specification activity consists in elaborating a representation of the artefact until it is so precise and detailed that the implementation of the artefact is completely specified. This implementation (realisation, manufacturing, fabrication, construction) of the specifications that result from design is considered a different activity from design —even if "design never ends", a statement that holds, as considered from many viewpoints. In this section, we will discuss it only with respect to implementation; later sections will discuss other design prolongations, especially through maintenance and users' participation in design both before and during its use. Even if the design project has been declared finished (the plans are transmitted to the workshop), its implementation may still require decisions. People realising the implementation task will often take them. From a cognitive viewpoint, these decisions are characterised as "design" (cf. our analysis of design as defined by the type of cognitive activity, rather than the type of task involved, or the status of the "designer").
NOTE: There is, in our opinion, a misunderstanding in the literature surrounding the term "artefact" as the entity to be designed. As we use this term —a rather common use in cognitive design research—, it refers to any entity that one may create, i.e., material or immaterial (music, software, social welfare policies, or any procedure), rather than pertaining only to material objects (e.g., buildings or machine-tools). The antonym of this "artefact" is a "natural" —not an "immaterial"— entity.

4.6 Design problems

As stressed already, the notion "design problem" doesn't convey any presupposition with respect, neither to the client's design specifications —one of the first forms that a design problem may take— being exhaustively and unambiguously defined or already being structured, nor to "the" problem representation being constructed once and for all.
The characteristics of design as a problem discussed in detail below are the following. Design problems are ill defined and complex. The constraints that govern a design problem generally are unstable, and more or less ambiguous. Design problems admit several more or less satisfactory solutions, not just one, correct and/or optimal solution. Finally, the solutions to design problems have a more or less direct impact on people.
We close this section in which design is characterised as a type of problem, presenting some characteristics of design problem solutions, especially some elements concerning the quality of solutions to design problem. Few studies have examined this clearly fundamental aspect from a cognitive perspective. With a few exceptions, it is still entirely the affair of design engineers and methodologists.
Before these different characteristics are going to be detailed in this section, we will present several problem typologies characterising design problems relative to other types of problems.

4.6.1 Design problems in problem typologies

Cognitive psychology classically distinguishes design problems from transformation and structure induction problems. Two other problem distinctions that are orthogonal to the previous one, are based on the partitioning into "adversary" and "nonadversary" problems, and on the opposition between "semantically rich" and "semantically impoverished" problem domains. In Artificial Intelligence, still other classifications are adopted. Two of them will be discussed hereunder: the "interpretation" vs. "generation" problems typology, and the opposition between "analysis" and "synthesis" problems.

The classical cognitive psychology distinction transformation, structure induction, and arrangement problems. The distinction in cognitive psychology that nowadays is considered the "classical" problem typology, distinguishes transformation, structure induction and arrangement problems. It dates back to the nineteen-seventies (Greeno, 1978). More recently, design problems have often been presented as constituting the third category —instead of, or together with arrangement problems (Greeno & Simon, 1988). Certain routine "design" problems can indeed be analysed as arrangement problems.
Other authors have refined or extended this classical typology. In his discussion of nonadversary problems (see below), Mayer (1989), e.g. proposes two other types of problems, deduction problems, and divergent problems. For Greeno and Simon (1988), however, deduction is reasoning, rather than a problem-solving task.
Greeno (1978) presents his three types of problems as ideal types, and notices that most interesting problems include components of different types. "Design" and "invention" problems are examples of such mixed types of problems: they may be analysed as problems that require "inducing structure in arrangement problems" (pp. 263-264). For Greeno, "the most demanding intellectual problems" are "composition" problems, which involve "[creation of] an arrangement of ideas whose structure incorporates some significant new understanding" (p. 264). Several studies, some of painters and Reitman's famous research on musical composition (1965) are referred to as indicating that "most of the intellectual effort involved in composition concerns defining and developing of problems, rather than solving them" (Greeno, 1978, p. 264) (cf. our discussion concerning problem "solving" vs. other problem-related activities). From our point of view, these characteristics fully apply to design problems in general, not only to musical composition.
In cognitive psychology, transformation problems are the types of problems that have been receiving by far the most attention. This holds particularly for Simon and colleagues, working from the SIP viewpoint, who examined transformation problems in the form of play problems (such as the Tower of Hanoi and other classical laboratory problems), rather than as problems solved in professional settings.
Discussions by the above-mentioned authors, of this problem classification (and its minor variations) that constitutes the reference in cognitive psychology, do not inform us much about the specificity of design problems. Some authors notice a difference between design and transformation problems that, if elaborated upon, might reveal relevant distinguishing features. According to Greeno and Simon (1988), in design problems, it is mainly the goal state that is constrained, whereas in transformation problems, initial and goal states are explicitly given, while there are constraints on the actions that may be used in achieving the goal state. We will elaborate these differences below in the section on design problems' ill definedness.

outine vs. nonroutine problems. The dimension "routine"-"nonroutine" is commonly used in design studies, but more in the domain of Artificial Intelligence than of cognitive psychology. Researchers working in the domain of Artificial Intelligence and Design (Gero, 1991; Gero, 1992; Logan & Smithers, 1993) (Gero, 1998; Gero, 1998) have often been making this distinction. Terms also encountered in this literature are "creative", "insightful" or "innovative" design, without the underlying precise distinctions always being clear (Brown & Chandrasekaran, 1989; Logan & Smithers, 1993; Navinchandra, 1991).
Brown and Chandrasekaran (1989), who focus on the construction of a theory of routine mechanical design, contrast "open-ended", "creative" design, and "routine" design —but, as they notice, the term "routine design" is "not meant to be formal or rigorous" (p. 35). Open-ended creative design is characterised by ill-specified goals and, as noted above, the absence of a "storehouse" of effective decompositions and of design plans for subproblems.
In *Exploration and innovation in design. owards a computational model*, Navinchandra (1991), opposes "routine" and "non-routine" design. Gero, in his "Introduction" to Navin-

chandra's book defines "routine design" as "design where both the available decisions and the computational processes used to take those decisions are known prior to the commencement of a design", whereas, in "non-routine design", "the available decisions are not all known beforehand" (p. vii). Navinchandra adds that "a design process is deemed routine if it involves a well understood sequence of steps where all decision points and outcomes are known a priori —there are no surprises.… Routine design systems can be organised in several different ways. For example, (1) some routine design tasks can be broken into discrete steps, where each step performs some specific design sub-task; (2) in some cases, the problem can be approached hierarchically, where each level in the hierarchy is predetermined; and (3) in other cases, specific heuristics are available for different parts of the design". (1991, p. 2)
Navinchandra's interest is in "innovative conceptual design". "Simply put, something is innovative if it solves a known or a new problem in a way different from other known designs". (p. 3).

Cognitive psychologists' distinction between routine and nonroutine problems refers to the knowledge that a person may bring to the problem. Mayer (1989) calls "routine problems" "familiar problems that, although not eliciting an automatic memorized answer, can be solved by applying a well-known procedure. Although the problem solver does not immediately know the answer to a routine problem, the problem solver does know how to arrive at an answer. For example, the problem 888 x 888 is a routine problem for most adults. In contrast, nonroutine problems are unfamiliar problems for which the problem solver does not have a well-known solution procedure and must generate a novel procedure". (p. 40)
NOTE 1: We recall that several authors in the SIP tradition (Hoc, 1988; Richard, 1990) consider that if, for a person, the representation of a task evokes a procedure allowing to attain the goal, the task constitutes no "problem" for this person.

NOTE 2: We stated already that many activities of professionals who are not considered "designers", would be qualified "design" when they are analysed from a cognitive viewpoint. Here, we want to add that many professional tasks that are qualified "design", involve activities that, analysed from a cognitive viewpoint, would not be qualified as such (see the discussion of design as a type of cognitive activity). Such tasks are sometimes qualified as "routine design", both by design professionals and by researchers in the domain. If one adopts this terminology, then, from a cognitive viewpoint, design activities are mainly implemented in "nonroutine" or in "creative" design (see Logan & Smithers, 1993).

NOTE 3: The distinction between routine and nonroutine problems is related to that between schemata and specific knowledge representations (that we proposed to consider as two forms of "problem/solution associations" in Visser, 1991b).

Semantically rich vs. semantically impoverished problem domains. A distinction that is quite different from the previous ones and that is used since the days that more realistic, or even real problems are being tackled in psychology and in Artificial Intelligence, is based on the nature of the problem domains. It contrasts "semantically rich" (or "knowledge-rich") and "semantically impoverished" (or "knowledge-lean") domains (Bhaskar & Simon, 1977). In semantically rich domains, problem solvers need much specific domain knowledge for the construction of problem representations. In order to solve problems in such domains, they use domain-specific problem-solving procedures, i.e. "strong" methods, in addition to general ("weak") strategies such as means-end analysis. In semantically impoverished domains, in addition to the information provided in the problem specifications, problem solving "only" calls for general knowledge and "weak" methods.
Design problems will generally pertain to semantically rich domains —except if a design problem is to be solved by a problem solver without any knowledge in the domain, that is, if the problem situation is restricted, probably in an artificial way, and, thus, impoverished. Indeed,

the distinction "semantically rich vs. semantically impoverished domains" depends for a large part on the knowledge that a person may bring to a problem. Domains that are generally considered as "semantically impoverished", such as that of artificial puzzles (e.g., the Tower of Hanoi, or the missionaries and cannibals problem), may become "semantically rich" for experts in that particular domain. They may bring to the problem many specialised and diversified knowledge in the domain that normally is exhaustively characterised by the problem specifications.

Artificial Intelligence classifications□interpretation vs. generation problems and analysis vs. synthesis problems. Two classifications proposed by Artificial allocate a particular place to design problems: interpretation vs. generation problems, and analysis vs. synthesis problems.

□Interpretation vs. generation problems. Hayes-Roth and Hayes-Roth propose a particularly interesting distinction between generation and interpretation problems (1979). The authors made an exceptionally rich analysis and model of a particular generation-problem solving, i.e. route planning (design of route plans) (1979).
Interpretation problems are "problems which present the individual (or computer system) with the lowest level representation of the problem content (e.g., the speech signal) and require interpretation of the highest level representation (e.g., the meaning)". (Hayes-Roth et al., 1979, p. 382) Generation problems are "problems which present the highest level representation (e.g., the goal) and require generation of the lowest level representation (e.g., the sequence of intended actions)". (Hayes-Roth et al., 1979, p. 382) "Interpretation and generation problems differ in important ways. For example, interpretation problems lend themselves well to initial bottom-up strategies, while generation problems lend themselves well to initial top-down strategies. Interpretation problems generally permit only one (or a small number) of solutions, while generation problems permit an arbitrary number of different solutions. Further, interpretation problems typically have correct solutions, while the correctness of solutions to generation problems varies under different evaluation criteria". (Hayes-Roth et al., 1979, p. 382) According to these distinctions, design problems are clearly generation problems. The qualification proposed by the authors presents two of the characteristics that we are proposing for design problems (different solutions, that are not "correct" or "incorrect", but dependent on the criteria adopted for their evaluation).

□Analysis vs. synthesis problems. A second classification that seems relevant in the present context is the one presented by, e.g., Clancey (1985) and the developers of KADS (see Breuker et al., 1987).
This classification of problem solving tasks is based on the concept of "systems" and their characterisation in terms of inputs and outputs. Along these lines, Clancey distinguishes synthesis and analysis as the two large sets of "generic operations" that may do things to, or with a system: they can construct a new system (synthesis) or interpret an existing system (analysis). Identifying design as a construction, that is synthesis task, Clancey distinguishes two types of design tasks, configuration (characterising a structure) and planning (characterising a process).
NOTE 1: We are not aware of configuration tasks having been studied from the point of view of cognitive psychology or cognitive ergonomics —except if one analyses the travelling-salesman problem as a configuration task. Given our idea of configuration tasks generally calling for "routine" problem solving (in the sense of Mayer, 1989), we consider "design" an inappropriate qualification. In configuration tasks, indeed, the initial state, goal state, and legal operators are, generally, all three well specified.
In the KADS knowledge elicitation methodology (Breuker et al., 1987), the top-level distinction between analysis and synthesis tasks is also adopted, with modification tasks as a transition area between the two. According to the type of input or output to the task, a finer classification can be established. Four types of design are so distinguished by the KADS authors: hierarchical, transformational, incremental, and multi-stream design. For synthesis tasks, the authors adopt a

two-stage model. As Simon, they distinguish (i) a stage in which an informal problem statement is being analysed, which results in a formal specification of the structure to be synthesised, and (ii) a stage in which this structure is effectively being created.
NOTE 2: This distinction between analysis and synthesis problems may be related to the ASE paradigm that underlies various engineering design methods. In these methods, design is, however, not either an analysis or a synthesis problem, but involves, in two consecutive stages, the problem solving activities corresponding to both types of problems.

4.6.2 Ill□definedness

As noticed already, I want to place on the ill-defined character of design another emphasis than did Simon (who, in his 1973/1984 paper, used the term "ill-structured ") (and so did Newell, 1969, who in addition adopted "ill formed").
NOTE: We adopt the terms "ill-defined" and "well-defined", because we judge the characteristic as not only referring to a problem's *structure*, but also to the more or less well defined character of the terms used in the problem statement (see their "vagueness" advanced by Newell, 1969, or their "open-ended" character).
The notion of a "well-defined problem" was proposed originally, as far as we know, by McCarthy (1956, as quoted by Newell & Simon, 1972, p. 73). Newell and Simon refer to McCarthy when they declare that "a problem proposed to an information processing system is *well defined* if a test exists, performable by the system, that will determine whether an object proposed as a solution is in fact a solution" (p. 73). The authors add that, for them, "performable" means "performable with a relatively small amount of processing effort" (ibid.).
Another author introducing the term is Minsky (1961). In his paper, Minsky never uses the term "ill-defined"; he only defines the "well-defined" version. We have to do with a "well-defined" problem, if "we are given some systematic way to decide when a proposed solution is acceptable" (p. 9). The author thus exclusively refers to a problem's solution, namely, to be precise, to its evaluation criteria.
Some years later, Reitman (1964; 1965) provided the first extensive discussion of "ill-defined" and "well-defined" problems. Adopting a SIP approach, he distinguished different types of problems, in terms of the three-component vector [A, B, ->] associated with a problem requirement. The first element, A, corresponds to the "initial state or object", the element B corresponds to the "terminal state or object", and -> corresponds to "a process, program, or sequence of operations" (1964, p. 284). Reitman proposes that the differences between types of problems depends on A, B and -> being specified more or less. Generally, in design problems, only B, the goal state, is the component that receives some explicit specification, even if poorly. Reitman applied his analysis to the processing of a particular design problem, i.e. the composition of a fugue (Reitman, 1965).
In his case study of an architectural design problem, Eastman (1969; 1970) states that he "extends" "the information processing theory of problem solving… to include ill-defined problems" (1969, p. 669). The problem studied by the author is a small-scale space-planning problem that consists in the selection and arrangement of elements in an existing bathroom that is to be redesigned. Even if this problem is not an extremely "ill" defined problem, Eastman presents interesting observations characteristic of ill-defined problem solving. The author asserts that "the significant difference between well- and ill-defined problem solving is shown to be a specification process similar to information retrieval processes" (ibid.). For Eastman, ill-defined problems are problems that lack (i) part of the problem specification and (ii) a formal representation language. "Most such problems also lack a precise formulation of an acceptable goal state". (1969, p. 669) In addition, many criteria that the specification must satisfy are left implicit (1969, p. 670). "Ill-defined problems are subjectively specified". (Eastman, 1969, p. 669) However, Eastman agrees with Simon, "the search processes used by humans to solve both [well-defined and ill-defined] problems [are] similar". (1969, p. 669)

Several authors seem to consider the notion "ill-defined problem" as a synonym of "design problem". This would mean, on the one hand, that design problems are characterised exhaustively by their ill definedness —whereas we consider that they are not. On the other hand, it would mean that there are no other ill-defined problems —whereas we consider that there are. Often these are "real-world problems", as they are called in the problem solving literature (Thomas, 1989) —even if this probably is neither a case of bijection: there are presumably real-world problems that are not ill defined. Examples of other ill-defined problems presented in the literature are the management problems mentioned by Newell (1969), the political science and other social sciences problems analysed by Voss, Greene, Post and Penner (1983), the judgmental and decision-making activities mentioned by Reitman (1964), requirements development as brought up by Carroll (2000), and the formulation of specifications as examined by Visser (1988a; 1990; 1988).

The analogical-reasoning problems whose solving is studied by Gick and Holyoak (1983; 1980) are considered "ill-defined" by the authors "in that the permissible operations that might be used to achieve the goal are left very open-ended. As a result, it is not immediately obvious how to apply a means-ends strategy, making it more likely that an available base analog will be perceived as useful". (Holyoak, 1984, p. 214) Because these problems are "ill-defined", they allow "a variety of potential solution plans" (1984, p. 215).

One may claim that several of the examples presented above might be analysed as design problems —a perspective that we have adopted in the past (Visser, 1990). On the other hand, although our definition of ill definedness is very broad, we consider that, in addition to this ill-defined character, design problems have other important characteristics.

Starting with Minsky, design problems have generally been qualified as "ill-defined" based on the ill-defined character of one or more of their components. Some authors consider design problems to be "ill-defined" because they admit several solutions, others because there are open constraints in the problem specification. I consider a design problem to be ill-defined with respect to the nature of all three problem solving components —i.e. the initial specifications, the operators that may be applied to solve it, and the goal state. This is the approach proposed initially —as far as we know— by Reitman (1964).

As noticed by Reitman, the goal state of a design project, if specified, is usually so at an abstract level, by way of the artefact's function and of constraints on this artefact. The initial state and the operators are generally underspecified. Implicitly, they may be supposed to correspond to the state of the art in the domain and/or the relevant knowledge that the designers possess. These "state of the art" and "relevant" knowledge are, for their part, clearly ill circumscribed.

Notice that these elements are central in making problem definedness a relative characteristic (like the "problematic" character of a task or situation). Definedness, indeed, depends on the data available concerning the task, i.e., it depends on the state of the art in the domain under study and on the knowledge possessed by the problem solver (covering also the problem-solver's knowledge of this state of the art).

As noticed by Schön (1988, p. 184), "different designers construe the task they are asked to perform in very different ways, and their different readings of the task lead them to very different global patterns of designing".

This view that design problems are ill defined with respect to all three problem-solving components, is also being adopted by Thomas and Carroll (1979/1984, p. 222). For these authors, "design is a type of problem-solving in which the *problem-solver* views his/her problem or acts as though there is some ill-definedness in the goals, initial conditions, or allowable transformations". The same relativity as for the "problem" character of a task, holds for the "ill-defined" character of a problem —and, as we will see in the Conclusion, even the "design" quality of a problem.

Ill-definedness is a characteristic that is relative with respect not only to the problem-solver, but also to the type of problem. Ill-definedness is a dimension: at one extreme extremely ill defined management problems and at the other extreme routine design problems. In the transition zone,

one may find, e.g., innovative engineering design problems, and standard architectural problems (row houses).
Taking this approach to its extreme, one may consider (as does Hoc, 1988) that, given the nature of a *problem* —i.e., in his view, a task that cannot be executed by simple memory retrieval— each problem is, by definition, ill defined. Its initial representation cannot trigger in memory an execution procedure. Its comprehension is by definition progressive (Hoc, 1988).
As we saw above, Simon's SIP companion, Newell (1969, p. 375), also considers problem definedness a relative characteristic. At first analysis, for Newell problem definedness depends on a problem-solver's knowledge of problem solving methods. "A problem solver finds a problem ill-structured if the power of his methods that are applicable to the problem lies below a certain threshold" (Newell, 1969, p. 375). A particular design brief may "look" ill structured to a designer who "has only [his] general problem-solving abilities to fall back on" [Newell, 1969 1524, p. 375). These same design specifications may "look" well structured to another designer whose experience and/or further knowledge leads him to evoke procedures for dealing with them. We saw also that Newell largely nuances his position, stating in the *Conclusion* of his paper that "the items… discussed [in his *Difficulties* section] —other aspects of problem solving… and the concept of vagueness— do not exhaust the difficulties or deficiencies of the proposed hypotheses. But they are enough to indicate their highly tentative nature". (p. 412)
In a discussion of "planning societal problems" already referred to above, Rittel and Webber have proposed a notion that seems very close to that of "ill defined", i.e. "wicked", which the authors oppose to the "tame" or "benign" character of problems (1973/1984). "Planning problems are inherently wicked". (Rittel & Webber, 1973/1984, p. 136) The authors oppose the social or policy planning problems to problems in the natural sciences. As planning problems depend on —political— judgment, they are never completely "solved", or solved once and forever (Rittel & Webber, 1973/1984, p. 136). A characteristic —presented as the seventh on a list of at least ten "distinguishing properties", others of which will be presented below— is that "every wicked problem is essentially unique" (Rittel & Webber, 1973/1984, p. 141). There are no *classes* of wicked problems in the sense that principles of solution can be developed to fit *all* members of a class. Despite seeming similarities, one can never be *certain* that the particulars of a problem do not override its commonalities with other problems already dealt with.

We conclude this section on design problems' ill definedness, by a brief mention of four characteristics of these problems that are related to their ill definedness.

- A factor contributing to the ill-defined character of design problems that is seldom referred to in the research literature, is the important role of clients' and prospective users' involvement in requirements development (Carroll, 2000). This factor affects the ill definedness of both design problems' "initial" state (the artefact's requirements evolve) and their goal state (by consequence, its purpose and functions also evolve). One may also suppose that "ordinary", leak people such as clients or users, adopt other procedures in their design contribution than the "legal" problem solving operators that can be exhaustively listed.
- Design problems are often considered as allowing several, more or less "satisfactory" solutions (a characteristic that is a consequence of what Simon called "satisficing", and that we indeed consider as essential, see below). If this feature depends undoubtedly on the ill definedness of the design problem's goal state, it is also related to the other two problem components' ill definedness. Indeed, with an initial state being ill-defined, a problem can, on the one hand, be interpreted and thus solved in various directions, and, on the other hand —and that is the viewpoint adopted by Simon— admit so many "alternatives" that only "satisficing", no optimising, is possible. Besides, the ill-defined character of its "operators" also has as a consequence that an "identical" problem can be solved in divers directions.
- The ill-defined character of design problem operators has consequences at different levels for the solving of these problems. First, at a "local" level, even if designers have many pre-existing design methods available, this repertoire is not necessarily sufficient. Designers of-

ten have to "import" procedures from other domains, both technical and general, "common-sense" knowledge. In her Mountain bike attachment study, Visser, e.g. showed the importance of designers using common-sense knowledge in industrial design. In addition, even if designers locally use pre-existing strategies, at the organisational, i.e. global, level of the design activity, the articulation of strategies will not be pre-established (e.g., top-down, breadth-first proceeding). Design activity is organised opportunistically, as has been shown in various empirical studies of design (this feature of the design activity will be discussed in detail below).

- In his presentation of arguments that challenge the SIP model, Mayer (1989) also questions the "atomization" premise of the model. By this label, he refers to problems having clearly delimited, i.e. well defined, "atoms", i.e. states and operators. As noticed by Mayer, for ill-defined problems, however, "determining the atoms… is often the crucial part of problem solving rather than the starting point. Further… the atoms may change as they are combined during the problem solving process". (p. 54)

NOTE: It is not obvious to make explicit or even clear, the differences between all the qualifications advanced by divers authors to characterise attributes of design "associated with" its ill-defined character. In addition to "ambiguous", these are qualifications such as "satisficing" (Simon, 1999), "vague" (see Newell, 1969, pp. 411-412), "wicked" (Buchanan, 1990, Spring 1992; Rittel & Webber, 1973/1984), "real-world" (Thomas, 1989), and "open-ended" (Brown & Chandrasekaran, 1989; Holyoak, 1984).

4.6.3 Ambiguity

Design problems are generally ambiguous: different people will interpret the problem states differently and differ with respect to what they consider legal operators (to speak in SIP terms).
For Reitman (1964), the continuum ranging from well-defined formal problems to such ill-defined empirical problems as composing a fugue —the particular design problem to which he applied his analysis— is closely related to "ambiguity". For ill-defined problems, this ambiguity indeed translates the scarcity or even absence of agreement over a specified community of problem-solvers regarding referents of problem attributes, permissible operations, and their consequences. This means that, even in a specified community, solutions to ill-defined problems are only more or less accepted. A problem's "open constraints" are the source and locus of this ambiguity, inter-individual variability, and problem ill definedness. According to Reitman (1964, p. 292), these are "surely the most important in characterizing ill-defined problems". An "open constraint" is a constraint "whose definition includes one or more parameters the values of which are left unspecified as the problem is given to the problem-solving system from outside or transmitted within the system over time". (pp. 292-293) In Reitman's approach to problem solving, the attributes that define a problem's components (i.e. A, B, and ->) may be viewed as constraints on the problem solution, and therefore, indirectly, on the problem solving process (p. 291).

4.6.4 Comple□ity

Design problems are generally large and complex, requiring for their resolution a long duration that may stretch from several days to several years. In my view, the size of design problems isn't, however, their essential characteristic. More fundamental is their complexity.
The notion of "complexity" defined in cybernetics refers to the information content of a group of elements assembled in a system by way of relations. A system's complexity not only depends on the number of its elements, nor on their density. What mainly determine this complexity are the relations that exist between the elements and the unforeseeability of the types of relations.
In design problems, the big number of components (variables) and their interdependencies would require a definite decomposition of the problem in order to make its resolution more feasible.

Decomposition is a method advocated by most design methodologies. According to Simon, decomposition is a powerful problem-simplification method. Generally, as noticed already, such decomposition is, however, difficult to accomplish, amongst other reasons because the problem structure does not dictate it. It is often based on their experience that designers decompose a problem (Goel & Pirolli, 1992). Rarely, however, the sub-problems resulting from decomposing a complex global design problem are independent. So, even if a designer proceeds in solving a design problem by decomposing it into several sub-problems, many problem solving will still be required in order to handle the interdependencies between the problem components, e.g. to articulate the different solutions resulting from solving these sub-problems —i.e. to manage a design problem's complexity.

4.6.5 Constraints on design solutions

In the domain of Artificial Intelligence, several authors analyse and model design as constraint management and satisfaction (Feitelson & Stefik, 1977; Logan & Smithers, 1993; Stefik, 1981a, 1981b; Steinberg, 1987).

Cognitive ergonomists have also examined the use of constraints in design. Bonnardel (1989, 1992 □79) has analysed their exploitation in the context of design evaluation. Darses (1994) has examined constraint management in the context of design generation. Based on the analysis of empirical data collected on a computer-network design project (LAN, that is, local area network design), Darses concludes that, even if handling these interdependent variables plays a central role in design, the designer's activity is not exclusively one of constraint management. The author indeed notices that other cognitive entities are used in design: the two examples she mentions are actions plans and schemata (Darses, 1990a).

Most authors who have discussed this question judge that, among the three problem components, the goal state is especially constrained. Therefore, in order to guide their activity, designers would use the constraints that govern the problem's goal state. These constraints guide the "narrowing down" of the set of possible solutions (Greeno & Simon, 1988). Simon notices that "one of the skills that the professional designer acquires in any domain is discernment of which constraints to satisfy first so that the plan can be carried to completion with a minimum of revision". (Simon, 1992a, p. 134)

In his analysis of design in terms of "satisficing", Simon focuses on evaluation. In addition to the constraints that we qualified as "critical" (i.e. evaluation criteria), constraints may also play a generative role (Visser, 1996). We noticed above that it is only in the context of "special" types of design such as social planning, that Simon also considers generative constraints (e.g. "interestingness" or "novelty").

As we saw above, Reitman (1964, p. 292) considers "open constraints" the most important factor of design problems' ill definedness. Even if, formally, it is generally possible to transform an ill-defined problem into a well-defined one by closing all open constraints (Reitman, 1964, p. 293), such a transformation corresponds to a form of "premature commitment" that designers may well regret afterwards. Early constraining of variables (constraints) will indeed frequently lead to less than optimal, unsatisfactory design solutions.

The constraints that design problems are to satisfy are often conflicting. Different trade-offs between constraints may require "satisficing" in Simon's terms, resulting in different acceptable solutions. Combined with the complex, i.e. multidimensional and interdependent organisation of design problems, this leads to making the analysis, structuring and pruning of relationships between constraints into an essential aspect of design activity.

If not all, then at least many constraints are open to discussion, perhaps certain types more than others. So are constraints with social, political, economic, and/or legal qualities.

□*emporal and spatial constraints*. This section presents two (series of) studies concerning temporal and spatial constraints in design. The first one has compared spatial design with temporal

design; the second one has examined design associating spatial and temporal constraints, i.e. spatio-temporal planning.

Using spatial and temporal isomorphs of a problem, Carroll Thomas, Miller and Friedman (1980) compared spatial with temporal design. The spatial form of the design problem led to more adequate solutions than the temporal isomorph, and they were designed in less time. This difference vanished when they provided participants with representation aids (a matrix representation). Thomas and Carroll (1979/1984, p. 232) conclude to "a notably strong tendency for spatial problem statements to encourage the use of graphic representational aids that were not encouraged by the temporal problem statement".

One might want to conclude that temporal constraints in design are more difficult to handle than spatial constraints. The following studies show that things are probably more complex.

In the context of research on spatio-temporal cognition (Visser, 2002a), we have studied a particular form of design involving temporal and spatial constraints, i.e. the design of route plans (see the famous study by Hayes-Roth and Hayes-Roth 1979).

Route planning, by definition, imposes spatial constraints right from the start: items to be organised into a route plan are specified to be at particular locations in space. Planning in general is, by definition, a question of temporal constraints. This holds for all types of planning, whether the artefacts have spatial characteristics or not. Therefore, route planning is a design activity that has the particularity to combine temporal with spatial constraints.

Two studies have examined how people articulate these two types of constraints, according to their knowledge of the environment to be traversed (Chalmé et al., 2000, in press).

The results show that, if they are to design a route plan that has to incorporate a number of chores governed by both spatial and temporal constraints,

- people who know the environment take into consideration the spatial aspects of the problem —i.e. the specificity of this planning— before its temporal aspects; whereas people who don't know the environment take into consideration the temporal aspects before the spatial ones —as in all planning tasks;
- people who know the environment try to optimise the duration of the itinerary; whereas people who don't know the environment try to optimise its length.

When people who don't know the environment are in less constrained conditions —they are confronted with only spatially constrained chores, and they are provided with additional information about the road network— they tend to plan as their knowledgeable colleagues with respect to their tendency in preferential processing spatial and temporal aspects. Integrating the spatial and temporal aspects of the problems, however, continues to be difficult for these amended "novices".

Our studies thus show that these two types of constraints are processed differently, their status varies with people's knowledge of the environment to be traversed, and this disparity is not uniform. People who know the environment favour spatial aspects with respect to the order in which they handle the constraints pending on a route: they favour, however, the temporal aspects as an optimisation criterion. People who don't know the environment proceed in an inverse manner: they first handle the temporal constraints, but optimise the route on spatial criteria. Second, the integration of both types of aspects seems to be difficult: it seems to call for good knowledge of the environment that is to be traversed. Providing people who don't know this environment with abundant spatial information on this environment, doesn't necessarily enable them to proceed to such integration.

With respect to constraint processing, a "classical" result in ergonomics studies is that, confronted with a task, people, in their effective task, generally *add* constraints to those imposed by the prescribed task. Several examples were noticed in the route planning studies.

First, there is the case of implicit temporal constraints, whose inclusion in the route plan may be considered as added constraints. A more interesting example is the case of the constraint that the route be of the shortest length possible. This spatial constraint seems to have been applied by many participants, even when there were no (prescribed) temporal constraints pending on their

chores. Indeed, many participants minimised the distance between chore locations —and thus the total route length— by applying a "nearest neighbour" strategy. This strategy, however, seems to have been applied less widely than in "travelling salesman" problems presented as such (Best & Simon, 2000).
At first view, planning a route between chores without temporal constraints might be compared to an *implicit* travelling-salesman problem. In the route plan studies, people didn't always approach the planning problem in this way, however. This may be due to several reasons.
First, contrary to what characterises travelling-salesman problems, the minimal-length spatial constraint wasn't presented, either explicitly, or implicitly (in so far as implicitly presented information can be controlled). Secondly, being faced with a real city (even if only on a map), participants probably added constraints to their prescribed planning task, to elaborate a realistic route, by incorporating plausible temporal task attributes (for example, a theatre booking office only opens late in the morning). Thirdly, Best and Simon (2000) notice that travelling-salesman problem solving may be best explained by planners adopting a meta-strategy that combines global and local strategies (especially, in the form of the "nearest neighbour" strategy). In one of our route planning experiments, meta-strategies combining global and local strategies (not only the "nearest neighbour" strategy) were also observed among participants who were familiar with the environment to be traversed. The participants in the route planning studies who did not know the environment, may have come to adopt a comparable approach when their mental load was alleviated (i.e., when there were no temporal constraints pending on the chores in their route planning task).
All questions concerning the relative difficulty of processing spatial and temporal constraints and concerning their articulation and possible integration, are thus clearly not yet elucidated (see also below the paragraph on Designing in space versus time, in the section on Different types of design).

4.6.6 Several satisfactory□rather than unique□correct solutions

Design problems don't have unique, correct solutions. As noticed, this problem characteristic is related to their ill definedness and to the "satisficing" character of design activity.
The solution formulated by a designer is, generally, neither "correct" nor "incorrect". It may be more or less satisfactory ("satisficing") amongst various acceptable solutions, which may differ more or less. The "final" specifications for the artefact designed by a designer may thus be contested by a colleague or —what is worse— by the client, not especially for being incorrect, but because, adopting other criteria than those adopted by the author of the solution, another one might be defended. This eventuality presupposes that the design criteria neither can be ordered according to an objective importance or validity rank, unanimously accepted by all participants involved in the design project. Contrary to what holds for well-defined problems, for ill-defined problems there is neither "a definite criterion for testing any proposed solution, [nor] a mechanizable process for applying the criterion" (corresponding to the first characteristic of the list that Simon, 1973/1984, has established for these problems).
Different designers come up with different design proposals —i.e. more than one idea is formulated per designer, and designers formulate ideas that are different from those advanced by their colleagues. This has been observed in different domains, such as architecture (Eastman, 1970), mechanical design (Frankenberger & Badke-Schaub, 1999), software design (Malhotra et al., 1980), and traffic signal setting (Bisseret et al., 1988).
In software design, "there is no way to determine which design or program code is better than another" (Kitchenham & Carn, 1990, p. 275). One of the underlying factors is that different ideas about the users' future work with the system can lead to different design solutions, one of which cannot be considered "better than another" (Löwgren, 1995).
The different results of a design process, i.e. the design "solutions", are clearly also related to the different ways in which different designers represent a design problem, and to their different ways of proceeding.

At least half of the ten "distinguishing properties" that Rittel and Webber (1973/1984) have presented for "wicked" problems are, more or less directly, related to this characteristic that links design problems with their solutions.
The first of these distinguishing properties is: "There is no definitive formulation of a wicked problem" (p. 136). Each formulation of a wicked problem corresponds to the formulation of a solution. This means that problem understanding and problem resolution are concomitant to each other: the process of solving the problem goes along with the process of understanding the nature of the problem.
The second property is that "wicked problems have no stopping rule" (1973/1984, p. 138). This is mainly because there are no criteria for "sufficient" understanding and because there are no ends to the causal chains that link interacting open systems. A designer (a societal planner, in Rittel and Webber's application) stops for considerations external to the problem: he runs out of time, money, or patience.
"Solutions to wicked problems are not true-or-false, but good-or-bad" (p. 138) is the third property enunciated by Rittel and Webber (1973/1984). The fourth one is that "there is no immediate and no ultimate, definite test of a solution to a wicked problem" (p. 139).
Finally, the sixth property, clearly applicable, is described under the long title: "Wicked problems do not have an enumerable (or an exhaustively describable) set of potential solutions, nor is there a well-described set of permissible operations that may be incorporated into the plan" (p. 140).

4.6.7 Impact of design problem solutions

The fourth "distinguishing property" of "wicked problems" presented by Rittel and Webber (1973/1984), i.e. the lack of a definite test for a wicked problem solution (p. 139), makes that the consequences of a design solution extend over a long period of time and are of concern to many people.
"Anticipating impacts on human activity", e.g. anticipating "users' future work with the system" is one the "characteristic and difficult properties of design" as distinguished by Carroll (2000, p. 39). Indeed, "design has broad impacts on people. Design problems lead to transformations in the world that alter possibilities for human activity and experience, often in ways that transcend the boundaries of the original design reasoning" (Carroll, 2000, p. 21) (see, below, our discussion of "the temporal nature of the artifact" in the Conclusion section on Different types of design).
This characteristic is related to the possibility to test a design. Indeed, in addition to the absence of a "definite criterion" for testing any proposed solution (pointed out, e.g., by Simon, and by Rittel and Webber), the artefact that is "under design" can, by definition, not be tested. In order to be tested, it should have been implemented already. A consequence is that omissions, failures, and other problems often aren't noticed until it is "too late". Another consequence is that, as mentioned already above, design never ends (see also Rittel and Webber's remarks on social problem solving).
Design domains differ to this respect. There are domains in which simulation —mental or physical— constitutes a particularly useful tool for designers in order to test solutions. In certain domains, material prototype versions and maquettes of the artefact can fulfil this function. This holds e.g. for the design of HCI systems, where users may participate in the design process, not only with respect to the requirements, but also for testing prototypes (see below).

4.6. uality of design problem solutions

Until today, few studies have been examining the quality of design from a cognitive perspective (see also Détienne, Burkhardt, & Visser, 2003; but see Fricke, 1999). This topic remains mostly the affair of design engineers and methodologists (Jedlitschka & Ciolkowski, 2003a).
In one of the "German empirical studies" on engineering design (Frankenberger & Badke-Schaub, 1999; Pahl, Badke-Schaub et al., 1999; Pahl, Frankenberger, & Badke-Schaub, 1999),

Fricke (1999) examined the possible consequences of "varying levels of completeness and detail in the assignment" on goal analysis and search for solutions, and on the quality of the results (p. 418), analysing the design process in terms of Pahl and Beitz' stage model (1977; 1984). Nine designers who all had been taught design methodology were presented with "two differently formulated, but in principle identical problems" (ibid.). "Five designers were given an extensive and precise assignment…. Four designers were given the problem in an incomplete, imprecise formulation". (ibid.) "Against [the author's] expectations,… there was no relationship between the type of problem formulation [precise or imprecise] and the total design time, despite the differences in time used for the goal analysis. Neither could any relationships be found between the solution quality and… duration of the goal analysis". (Fricke, 1999, p. 422) "Only the comparison of qualitative characteristics of the approaches with design quality gave further information". (Fricke, 1999, p. 422) This comparison led the author to the following conclusions (pp. 428-429):

The four "good", i.e. successful designers (two in each problem formulation condition)

1. clarified problem requirements, focusing on problem structure;
2. proceeded to "critical analysis", "actively [searching] for information, critically [checking] given requirements and [questioning personal] requirements regarding their priority";
3. "summarised information of the problem formulation into requirements and partially prioritised them";
4. "did not suppress first solution ideas even in the goal analysis", but also "repeatedly tried to return to 'clarification of the problem' until this phase had been completed";
5. "detached themselves during the conceptual design stage from the pre-fixation generated during the goal analysis";
6. "produced variants, but kept an overview in that they alternately generated solutions (blockwise), and then assessed them consciously to reduce the number of suitable variants";
7. "possessed advanced engineering knowledge, which resulted in better technical assessment".

Considering that "the designers who produced good solutions used approaches that encourage success" (p. 428), Fricke proposes to draw, from these results, consequences for design methodology. The observed detachment from specific solution ideas during the conceptual design stage (point 5), e.g., leads him to suggest that "solution-neutral formulation of requirements seems to be an important pre-requisite" for successful design. Based on point 7, the author concludes that "design methodology… does not make up for a lack of engineering knowledge, but can assist in obtaining good solutions if it is applied flexibly in accordance with the problem to be solved" (p. 429).

In the engineering design domain, Hubka and Eder (1987) consider four design factors as mainly influencing the quality of engineering design: the design engineer, the working means, the knowledge and/or know-how (methods, technical information, representation methods, management), and the working conditions and/or environment. In a matrix, the authors represent the influence that each factor has on a number of design process characteristics (quality of technical system, duration of design process, efficiency of design process, risk of failure, transparency of design process, non-routine designing, cost of the design process, and quality of description) (Hubka & Eder, 1987, p. 132). Without presenting further underpinning, the authors assert that design engineer and working means are most influential, whereas working conditions and/or environment have only small or even no influence on the design process.

In the domain of software design, many measures have been proposed for the assessment of software quality (Jedlitschka & Ciolkowski, 2003a), but they have rarely, if ever, been the objects of cognitive evaluation studies (Détienne et al., 2003).

In order to present a discussion of organisational factors influencing the quality of design, Winograd proposes, in his series of essays *Bringing design to software* (1996), Norman's famous "war story" on the complications of "something as simple as the placement of [a] power switch" on the Apple computer (as qualified by Swaine, Retrieved 27 August 2003, from

http://www.ddj.com/print/documentID=13370). Winograd introduces Norman's paper as an analysis of the kind of culture or organisational structure that promotes effective design. Norman reports in detail "how a dedicated committee tried to simplify the placement and function of the switch, but succeeded only through multiple compromises in the face of many reasonable technical problems" (Norman, 1996, p. 234). What Norman, however, especially describes in great detail, is how the Apple organisational culture makes design a complicated matter, even the design of the power switch on the consecutive exemplars of a series of Apple computers. He doesn't, nevertheless, give any precise, general answer to the question about the role of the organisational structure. He presents the enormous number of compromises to be attained between a big number of people who come from different backgrounds, and thus have different viewpoints on the design and hold different stakes in it. One of Winograd's concluding remarks to this respect is that "an organization can do much to support the process and ensure success", but it can also "impede the design process, stifle creativity, and damn a project to software hell".

4.7 Design activity

This section presents some characteristics related to design as an activity. In the introductory paragraph, we have already defined this activity as "specification" of representations of the artefact that is the object of design. This activity thus involves generation of representations (the specifications for implementing the artefact), and it is therefore that knowledge and representations of different types play a particularly important role in design.
At least two among the definitional cognitive aspects of design as an activity that were noticed by Simon are particularly important in our approach to design. These are Simon view of design as a type of cognitive activity rather than a professional status, and as a "satisficing" activity.
We further characterise design by its iterative and, especially, opportunistic character. The central activities in design, i.e. construction of problem representations, solution generation and solution evaluation, are not limited to a particular problem solving stage; nor aren't they implemented once and for all in a particular stage or phase that occurs at a particular moment in the global design process: these design activities occur all along the process.
Another characteristic of design introduced and discussed below is its creative character. Few authors have insisted on this aspect, whereas it constitutes in our view a defining feature in the "family" of design characteristics (Hesse, 1988).
We will conclude this section by the discussion of a number of typical, if not specific, design strategies: planning and organisation, decomposition, reuse, and user considerations guiding development and evaluation.

4.7.1 Design as a type of cognitive activity rather than as a professional status. Some additions to the SIP approach

We concurred above with Simon's idea that design is a type of cognitive activity, not a professional status. Simon may propose a somewhat large definition of design when he writes that "everyone designs who devises courses of action aimed at changing existing situations into preferred ones" (Simon, 1999, p. 111). We do not consider that, e.g., physicians prescribing remedies for their patients, are designers —their main cognitive activity will rather be analysed as one of diagnosis (i.e., a kind of structure induction type of problem solving). However, their medical colleagues devising a new therapeutic protocol, e.g. for treating cancer, may be considered to be involved in a design process. In the same way, a professional who is applying laws will not be considered a "designer", but the legal specialists drawing up new laws will.
The viewpoint according to which other people than so-called "designers" are involved in activities that, from a cognitive viewpoint, are analysed as "design", has another side. Indeed, not each professional considered as "designer" —by management and colleagues— is constantly involved in what we consider "design". Indeed, as noticed already, not all professional tasks qualified as "design" involve much activity that, considered from a cognitive viewpoint, can be qualified as "design" (cf. our view of "design" as distinct from "routine design").

NOTE: authors who don't base this critique on a cognitive analysis of people's activity have advanced the idea that the term "designer" is being used inappropriately also. They judge that "the term 'designer' is applied too narrowly" because the qualification ignores certain participants in the design project who "may be affecting the course of the design as much as any of the designated 'designers'" (Burns & Vicente, 1995, p. 102).
Schön also proposed to broaden the notion of "design professions", showing how professionals of many disciplines create what he called "design worlds" "in the spirit of Nelson Goodman's Ways of Worldmaking" (Schön, 1992, March, p. 4). Design worlds are "environments entered into and inhabited by designers when designing" (Schön, 1988, p. 181). The "indeterminate zones of practice" in which many professional practicioners are working, "especially… situations of uncertainty, uniqueness, and conflict", may be qualified as "design-like" (see also Bucciarelli, 2002; Schön, 1988, p. 181).

4.7.2 Design as "satisficing" both in generation and in evaluation of solutions

As noted already in several contexts, we completely agree with Simon analysing design as a "satisficing" rather than an optimisation activity. We stated, however, that satisficing does not only occur in evaluation, but also in generation of solutions. Eastman (1970) observed these two forms of satisficing —whereas optimisation was rare in his study. "Only one instance of an attempt to optimize a solution has been found in thirteen protocols concerning three different design tasks". (p. 30)

4.7.3 Creativity

In an everyday acceptation, "creative" is often taken as a qualification of a product, the result of an activity. Cognitive psychology, in its qualification of design as "creative", considers the activity rather than its result. We use "novelty" for a particularly new, original result —that may be the outcome of creative activity or not.
Considered from our perspective, "creative design" may have, at least, two denotations. It may be a synonym —the explicit denomination— for what, in this text, is considered "design", that is, design as analysed from a cognitive viewpoint. It may also refer to a specific form of design that is particularly "inventive". This form has been discussed already, in passing, in the section "Underestimating the role of 'nondeterministic' 'leaps'". This section mainly discusses the creative aspect of design in the first sense.
The creative character of the design activity is an essential characteristic that has, however, not been proposed by many authors in the domain of cognitive psychology and cognitive ergonomics. In Archer's definition of design, it is an essential component: "Design involves a prescription or model, the intention of embodiment as hardware, and the presence of a creative step". (Archer, 1965/1984) "Arriving at a solution by strict calculation is not regarded as designing". "Some sense of originality is also essential". (Archer, 1965/1984, p. 58) However, these last two characteristics are relative. "Just how much originality is needed to distinguish the preparation of working details from actual designing, and just how little is inevitably required to distinguish calculation from designing, is very difficult to prescribe, especially in the field of mechanical, electronic, and structural engineering. The satisfaction of these two conditions together is sometimes referred to as the presence of a creative step". (Archer, 1965/1984, pp. 58-59) (see also De Vries, 1994)
A consequence of this definition of design is that more design activity will take place in early than in later phases of a design project. The activity that is considered "design" in this text, i.e. design as analysed from a cognitive viewpoint, mainly occurs in what organisational or socio-technical approaches of design, based on prescriptive models, call the "early stages" of design.
Even if we consider "creativity" an essential characteristic of design, designing and artistic creating are, however, not the same activities. There is indeed a difference between "an architect preparing plans for a house" or "a typographer preparing a layout for a page of print" —both "clearly designing"— and "a sculptor shaping a figure" —who is not designing, according to

Archer (1965/1984, p. 58) —and we agree. Two necessary elements for qualifying an activity as "design" rather than "creation" are the "prior formulation of a prescription or model" —what we call a "specification"— and "the hope or expectation of ultimate embodiment as an artefact" (Archer, 1965/1984, p. 58). "When a sculptor produces a cartoon for his proposed work, only then can he be said to be designing it". (Archer, 1965/1984, p. 58)

4.7.4 Problem-representation construction, solution generation, and solution evaluation

From an analytical viewpoint, at a high level of abstraction, one may consider designing, i.e. the design process, as a sequence of different design problem solving "steps" implemented in order to solve the design project.

Following this approach, it may be modelled as actions (the "steps") that proceed in iterative cycles, and lead progressively from a globally specified problem (problem specifications) to a detailed implementable solution (specifications of the artefact). In each cycle, problem solving may be considered to proceed in three steps: problem-representation construction (shortly called "problem representation", if there is no risk of ambiguity, at least), solution generation (or "solution development", Visser, 1991b), and solution evaluation. Generation of a solution leads to the proposal of a solution. Each proposed solution is evaluated and may be accepted or rejected (possible finer distinctions are not used in this text).

In order to obtain a final, implementable solution, problems are worked out, simultaneously, in two directions. On the one hand, problems are worked out "in breadth", decomposed into sub-problems —which imposes intermediate solutions to be integrated at a certain moment. On the other hand, problems are worked out "in depth": each sub-problem is to be refined, on an abstract-concrete dimension, until an implementable solution is reached, i.e. the most concrete form of design solution. This means that each solution to a sub-problem constitutes a "problem to be solved" as long as it is not yet implementable, that is, not yet detailed and concrete enough to specify its implementation.

This analytical model, however, does not characterise the actual design activity, which progresses opportunistically, that is, doesn't proceed as a series of cycles that consist of actions occurring in a fixed order, such as representation - generation - evaluation. This opportunistic organisation is presented below. In this section, some remarks will be formulated concerning the two "steps" that are not detailed elsewhere in this text, that is, generation and evaluation (problem-representation construction has been discussed at several occasions throughout this text and will be the object of a special discussion in section "Using representations of different types: viewpoints and levels").

The way in which designers handle the "double" problem refinement, in breadth and in depth, is one of the factors that contribute to design being organised opportunistically. The section on "Decomposition in the SIP approach" discussed some of the difficulties involved in the refinement that proceeds "in breadth", transforming problems into sub-problems (they will be taken up below in the "Design strategies" section on "Decomposition").

The abstract-concrete solution specification ("in depth" refinement), which in mechanical design domains often transforms a design brief, from conceptual problem specifications, via structural specifications, into physical specifications (Visser, 1990), also proceeds less straightforward than presented by design methodologies (see also Darses, 1997). These methodologies suppose that, throughout a design project processing, sub-problems resulting from in-breadth refinement are tackled and maintained on the same "level", i.e., of the abstract-concrete axis, so that, at each moment in the design process, all problem-solutions under consideration are on the same refinement "level". Empirical design studies show that this is not the case at all: some problem-solutions already have a particularly concrete form, while other problem-solutions, co-existing with the previous ones, are still specified at a highly abstract level (Visser, 1990) (Darses, 1990; Darses, 1994).

With respect to solution generation, Visser (1991b) after presentation of the analytical model sketched above, identifies and analyses different types of processes that may implement this

activity: from the "simple" "evocation" of solutions existing in memory, to the "elaboration" of "new" solutions out of mnesic entities without any clear link to the current problem (cf. "nondeterministic" "leaps" and creativity). The distinction between "evocation" and "elaboration" is once again an analytical one that isn't absolute. Except when its basic solution constituents come from an external information source, the elaboration of a solution uses mnesic entities, which will have been activated, via an "unguided" activation mechanism ("evocation"), or by "guided" activation (two forms of activation distinguished in the Visser, 1991b paper). Visser (1991b) presents the different types of problem-solving actions as particularisation forms of these two types of modes, illustrating her presentation by observations on the way in which designers solve two mechanical aerospace design problems.

NOTE 1: Construction of problem representations may also adopt these two types of processes, i.e. "evocation" and "elaboration". Many studies in the domain of software design have been concerned with problem representation in its evocation form. The main framework for such analysis adopted the "schema" as knowledge representation format and "schema instantiation" as the "evocation" process (Détienne, 2002a).

NOTE 2: From slightly different viewpoints, another opposition between solution generation modes can be established. A first one is: generation by knowledge retrieval ("preparation" in Newell, 1990's terminology, through pattern-matching) (see our "evocation") vs. generation by search ("deliberation" in Newell, 1990's terminology) (see our "elaboration"). A second one is the opposition between "designing as search" (or "as planning") vs. "designing as exploration" (Gero, 1998, p. 3; Logan & Smithers, 1993).

Many studies have shown the intertwined character of generation and evaluation of solutions. As one of the characteristics of the opportunistic organisation of design, it will be discussed below.

Evaluation is generally characterised as an activity that, in the case of problem solving, confronts solutions to evaluative references.

Many types of such evaluative references ("constraints" or "criteria") and evaluation modes have been distinguished. Analysing composite-structure design tasks, in experimental and field situations, Bonnardel (1989) uses two types of distinctions.

According to their source, she distinguishes three types of evaluation constraints:

- prescribed constraints, which are inferred by the designer from the problem specifications;
- constructed constraints, which designers establish on the basis of their knowledge;
- deducted constraints, which designers infer based on other constraints and/or of the actual state of the design project (the problem-solution).

Bonnardel (1989) proposes a continuum on which different families of evaluative references are situated. She reserves the term "constraints" for operative evaluative references, and "criteria" for conceptual references. We have proposed to reserve "constraints" for generative references that steer solution generation, and "criteria" for critical evaluative references guiding solution evaluation (Visser, 1996).

As evaluative references are knowledge, one may expect that designers' expertise in a domain influences their reference use. This has indeed been observed by Bonnardel (1992) who concludes that according to their competence in the domain of the specific design task and to the routine character of this task, designers are able to bring more or less evaluative references to their task. D'Astous, Détienne, Visser, and Robillard (in press) have observed this in a field study of collective design, i.e. technical software review meetings.

Depending on the reference for evaluating a solution, different strategies (evaluation modes, Bonnardel, 1992) are used. The reference can be uniquely a constraint ("analytical" evaluation mode), but other designs (solutions) can be referred to as well ("comparative" evaluation mode) (Bonnardel, 1991b).

Before presenting our last observation concerning evaluation, we need to introduce briefly our global model with respect to design organisation, based on a blackboard approach.

We distinguish between an action-execution level where design problem-solving actions are "proposed" and "executed" and a control or action-management level where a "selection" is

made among the actions proposed; at this management level, control also makes "calls for action proposals". The two levels are structured according to the iterative sequence presented in Table 2 (see also below our discussion of The opportunistic organisation of design in our section on Planning and organising design).
Evaluation has functions at both levels. Its two classical functions are at the action-execution level of the design activity, i.e. assessment of design solution proposals leading to the selection of a solution proposal (see, e.g. Bonnardel, 1992). Evaluation has also a function at the action-management level, in that it affects the continuation of the design process (Visser, 1996). Designers, indeed, do not only evaluate solutions or evaluative references, but also their possible design actions (Visser, 1996).
NOTE. The present text focuses on individually conducted design. In collective design, solution proposals are being evaluated not only by their author, but also by colleagues. This "broader" evaluation causes several differences between these two forms of design (D'Astous et al., in press; D'Astous et al., 2001; Visser, 1993a).

Table 2. Articulating design actions at the action-execution and the action-management levels

(i) call for action proposals
(ii) proposal of one or more actions
(iii) evaluation of the proposed action(s), leading to selection of one action
(iv) execution of the selected action
back to (i)

4.7.5 Using knowledge of different types: domains and abstraction levels

Knowledge is one of the main resources used to construct problem representations, and to generate and evaluate solutions. This holds in all problem-solving domains, but in design, the role of knowledge is particularly critical. First, design is a "knowledge-intensive" type of problem solving activity that takes place in domains that are "semantically rich" for the problem solver. Solving a design problem indeed requires specific domain knowledge and procedures from various domains and of different abstraction levels. Second, design consists in construction of representations, until a "final" representation, i.e. the artefact's specifications. This second point will be discussed in the Conclusion section.
Concerning the first point, design projects generally require

- knowledge in one's specific design expertise domain (e.g. mechanical, electrical or civil engineering design knowledge; plus often more specific expertise, e.g. in mechanical design, aerospace design; or even more specifically, structures in aerospace design);
- knowledge in other, underlying technical and/or theoretical domains (e.g., physics, mathematics, or control theory);
- knowledge of design methods —or of at least one particular design methodology designers are supposed to follow;
- knowledge in the application domain, e.g. air shuttles (see Visser, 1991b);
- knowledge in non technical domains (see Visser, 1995b);
- often, at least some basic knowledge from ergonomics (because most artefacts will be used by people…);
- and sometimes knowledge concerning social, political, economic, and/or legal aspects of the artefact and its use.

As designers generally are expert in one specific domain of design expertise (e.g., structures in aerospace design), the need for knowledge from many different specific design domains leads to design projects often requiring the collaboration of specialists from various domains of expertise, i.e., it leads to large projects being mostly a matter of collective design. In mechanical engineering design, e.g., these may be structure, system, workshop, and maintenance.
With respect to abstraction levels of knowledge used, designers of course utilise much generic, abstract knowledge. However, the reuse of specific knowledge, i.e. knowledge linked to particular design projects, plays a role that is more important than has traditionally been assumed —and thus examined— in approaches to problem solving (see, e.g., Visser, 1995b). This strategy will be discussed in an independent section.

4.7.6 □sing representations of different types: levels and "viewpoints"

Independently from the need for different types of knowledge, solving a design problem also involves handling different types of representations. Once again, this characteristic that holds for all problem solving is particularly central in design problem solving, given that the object of design is construction of representations.
Both internal and external representations are involved. As representations are being constructed using knowledge, both types of representations are of course strongly related to a designer's knowledge, even if this holds most strongly for internal, i.e. mental, representations.
All through the design process, designers produce and use external, material representations. According to the domain of design, designers use textual and/or graphical drafts, flowcharts and other graphical productions (such as drawings and diagrams), but also textual annotations, memos, colouring, sizing, or positioning of objects (Nakakoji, Yamamoto, Takada, & Reeves, 2000; Newman & Landay, 2000). The reference, and even the meaning, of these representations may be vague and actually change over time. The processing of graphical productions may depend on perceptual rather than cognitive processes (ibid.). □hang and Norman (1994) pose that different types of representations activate perceptual and cognitive processes. "Perceptual processes are activated by external representations, while cognitive processes are usually activated by internal representations" (□hang & Norman, 1994, p. 118). We might add that these internal representations may be the result of processing external representations, be they graphical or textual.
One of the functions of external representations, the one that comes most easily to mind, is that of communicating, reporting, or saving a solution.
Another function of external representations has been qualified as "computational offloading", i.e. discharging internal working memory (□hang & Norman, 1994). On the basis of results obtained in a study that examined solving the Tower of Hanoi problem in different versions, i.e. governed by more or less rules that could be internal or external, □hang and Norman (1994) conclude that "computational offloading" through external representations —rather than making people "internalise" the corresponding rules— reduces the load on working memory, thus providing more "space" for planning subsequent moves (Rogers & Scaife).
In many design domains, particularly those that are concerned with physical artefacts, such as mechanical or architectural design, external representations serve still another, essential function: they are a reasoning tool for designers. Each design act produces —intermediary— results, often external representations, which may lead the designers, in a "reflective conversation with the situation" in which their activity is "embedded" (Schön, 1992, March), to evolve in their interpretations, intentions and ideas for solutions. Even if this characterisation rather applies to individual design, an analogous view holds for verbal and gestural interaction in team design. For the designers themselves, their design contribution using one of these representational modalities has a comparable function, that is, it constitutes a tool that enables and even helps them to elaborate their design solution. For the designer's partners, it plays an analogous role, allowing them to position themselves —and their solutions— with respect to the design proposal enunciated or represented otherwise (in graphical or gestural form).

The representations produced and used in early and other intermediary design phases are not necessarily of the type used as the final representation, i.e. the specifications of the design artefact specifying its implementation. Amongst other functions, intermediary design representations allow designers to focus on different aspects of their design (Newman & Landay, 2000).
NOTE 1: It is interesting to note that, contrary to most other types of external representations, speaking doesn't allow for drafts.
NOTE 2: In team design, verbal and gestural interaction constitutes another representation "medium" (see above).
NOTE 3: In this text, the notion "viewpoint" —recently adopted by many authors, but without clear definitions— will not be used with a technical meaning, as referring to a particular type of representation. It is often used to refer to the specific representations of design participants that are due to their specific professional knowledge, domain of expertise an/or know-how. However, this acceptation of "viewpoint" doesn't justify, in our view, the introduction of a term different from "representation": the representations referred to don't have characteristics that differ from those classically attributed to "representations".
With respect to the representation "levels", these may be decreasing "abstraction" levels (in terms of Rasmussen, 1979): from representations expressed in "concepts related only to the purpose of the system" to representations expressed in "concepts which are related to purely physical properties of the system" (where, in design, the artefacts correspond to Rasmussen's "system" under diagnosis). Representation levels of decreasing abstraction may be implemented in a mechanical design project, e.g., by functional, structural, and/or physical problem analyses (see, also Darses, 1997). As noticed already, something typical of design problem solving is that designers do not traverse these representation levels of decreasing abstraction in a methodical, fixed order. One of the characteristics of their activity is that designers move from a representation at level n to a representation at level n□m, i.e. they may proceed from a functional to a physical analysis of the problem, after which they may turn to a structural analysis, before handling again a problem on its functional level (Visser, 1990) (see also Darses, 1997).
Among the external representations of different representational formats, especially the graphical ones take on form in different "levels". In architectural or mechanical design, graphical productions generally differ depending on the design phase in which the designer is involved (for architecture, see Lebahar, 1983). In the initial phases, they are necessarily vague (in the form of sketches) in order to explore possibilities without focusing on, or even handling with, low-level details. In this way, designers leave open as many possibilities as possible, i.e. they maintain as many degrees of freedom as possible in their solution (cf. their intent to refrain from premature commitment). In each phase of a design process, problem specifications are anyway differently represented —be it internally or externally. This characteristic has often been noticed concerning the gap that has to be covered between the first specifications (those specifying the problem) and the final ones (those specifying the solution), but it is not restricted to these extreme phases of a design project.
In his study of architectural design, Eastman (1970) has observed "consistently" in all his protocols that designers, "instead of generating abstract relationships and attributes, then deriving the appropriate object to be considered,… always generated a design element and then determined its qualities". (p. 27) This author considered as the most general finding of his studies "the significance of representational languages to problem-solving ability" (p. 30). Depending on the types of representations they have at their disposition, designers dealt more or less easy with certain types of constraints. Eastman also has noticed a "difference in the constraints considered by those designers who worked in section versus those who did not.… Generally, a clear correspondence was found between the kinds of constraints that could be considered and the representations used". (p. 30).

4.7.7 Selecting and sticking to a "kernel idea"

Several authors in the domain of empirical design research have observed that designers, in an early phase of their activity, tend to select a "kernel idea" (Guindon et al., 1987; Kant, 1985; Ullman et al., 1988) and to stick to this "central concept" (Ullman et al., 1988) in what is going to become their global design solution. This idea functions as a "primary generator", concept introduced in 1979 by Darke (1979/1984) as referring to a few simple objectives that designers adopt at the very start of a project in order to generate the initial solution kernel —an approach also referred to as "position-driven" design.

Ullman, Dietterich, and Staufer (1988) studied how two mechanical design problems, each of a different type, were solved by experienced mechanical designers who could be expected to have a high degree of expertise for the selected problems. The authors conclude that the designers, rather than first to formulate a global design plan, develop and gradually extend "a central concept" (p. 36).

Kant (1985) makes a similar observation, noting the importance of what she calls a "kernel idea": design generally starts by focusing on an idea that is "quickly selected from those known to the designer… [who] lays out the basic steps of the chosen idea and follows through with it unless the approach proves completely unfeasible" (p. 1362).

Guindon, Krasner, and Curtis (1987, p. 64) have also observed that the designers they studied focused quite early on in the design process on a kernel solution based on a primary generator that directs the complete design process and all the generated partial solutions.

Exceptions, however, have also been observed. Some authors have supposed that individual and/or de-contextualised design may induce such early fixation on a kernel idea (Ball & Ormerod, 2000). Ball and Ormerod (2000) indeed note, that in their former studies of individual designers, they observed early solution "fixation" comparable to the results presented above. "There is evidence for satisficing in the selection and evaluation of putative solution options. Individual designers frequently become fixated upon single solution ideas (usually derived from prior experience) rather than exploring alternatives in order to optimise choices…, even when they recognise that the solutions are less than satisfactory". (p. 157) In the authors' 2000 study of design review meetings, however, they observe how "a design team [explores and critically appraises] a range of alternative design concepts in an in-depth manner". (p. 164) The results of study are considered "surprising" by the authors "in the degree to which they contrast so markedly with [their] previous studies of individual and de-contextualised designers…. [In their 2000 study], almost all episodes appear to reflect a motivated attempt by designers to generate and evaluate multiple solutions options… with the aim of maximizing the choice of a best alternative in relation to a particular design question.… This evidence for optimising attempts at multiple solution search and evaluation is a finding that generalises across the different companies we researched and the diverse design tasks that were being tackled within these companies". (p. 163)

It was the team manager who, in this Ball and Ormerod (2000) study, acted as a "safeguard against premature commitment to single-solution options should either the project champion or the group of collaborating designers become overly fixated upon a particular idea or concept prior to a full and cautious exploration of viable alternatives. It appears that the team manager has a very clear and explicit notion of the importance of ensuring multiple solution search and evaluation such that they are seen to perform an important steering function when designers get fixated upon single or unsatisfactory solution ideas or need support in pursuing alternatives". (p. 164)

In our study on a team of mechanical designers working on an unfurling antenna for a satellite (Visser, 1993a), we also observed a multitude of solution principles being advanced, but all formulated by two designers, the most experienced ones.

There are however also studies of individually conducted design that have observed designers to come up with several solution ideas. The industrial designer whose protocol is analyzed by Eastman (1969, design for a bathroom) doesn't seem to stick to one "kernel idea". He formulates five alternative solutions that he both generates and evaluates —even if only two are being com-

pletely developed. More in general, the designers studied by Eastman proceed right from the start by specifying design elements at a concrete, rather than abstract level.
In a study comparing CAD-using mechanical designers with manually drawing designers, both working individually, Whitefield (1989) observes that the drawing designers generate (that is, take into consideration) more solutions than the CAD users. The knowledge required to operate a CAD system appears to interfere with the application of the knowledge required to make domain decisions. Contrary to what propaganda suggests, CAD doesn't unburden or "free" its user: producing a drawing using CAD is a more complex and demanding task for a designer than drawing it manually.

Premature commitment. Selecting early on in a design process a "kernel idea", and sticking to this idea, may be considered a form of "premature commitment". Several authors in the empirical design studies domain notice, however, that designers often try to avoid premature commitment (Lebahar, 1983; Reitman, 1964, p. 293). The apparent contradiction may be removed in at least two ways:

- designers may declare, or have as their ideal, to refrain from premature commitment, but in practice not handle in consequence;
- early on in a design process, designers may select a "kernel idea" at a conceptual level, but after that, they may refrain from premature commitment at a more concrete level, by not fixing (all) values for its variables.

4.7. Design strategies

During the second half of the nineteen-eighties, several studies in the domain of "psychology of programming" have examined design strategies. They did so, however, mostly on students, that is, novices. In case they studied "experts", these were advanced students rather than professionals (see, e.g. the papers in the special *Le Travail Humain* issue on "Psychologie ergonomique de la programmation informatique", 1988).
There are, however, several studies identifying strategies used by expert, i.e. professional designers. In an experimental study of a simple, but realistic architectural design problem (a single-person dwelling) solved by a professional designer, Akin (1979/1984) observes several specific design strategies, which he characterises with the following names: (i) obvious-solution-first, (ii) divide-and-conquer, (iii) pre-compiled-solution, (iv) most-constrained-first. These four "unique forms of heuristic search" are used in addition to more familiar, general forms of search strategies, such as (i) hypothesise-and-test, (ii) induction, (iii) means-end-analysis, (iv) hill-climbing, and (v) pattern-matching.
The following five sections present some data on design strategies that have been observed in several studies to be used by professional designers, i.e. decomposition and planning, planning and organisation, reuse, simulation, and user-oriented design (see also Gilmore, 1990; Visser & Hoc, 1990).
NOTE: We first present planning in combination with decomposition, and afterwards with organisation. Indeed, decomposition of a task not only leads to a list of sub-tasks, but also —be it implicitly— to a plan specifying the order of dealing with them. With respect to planning and organisation, we will see that an action plan —resulting from a decomposition activity or retrieved as such from memory— is one thing, the actual organisation of the activity is another! Confronting the two has been the basis of our studies on opportunism (Visser, 1990, 1994a).

Decomposition and planning in design. As noticed above, decomposition is considered an important strategy in design by researchers working on design from another than a cognitive viewpoint (see our discussion of Simon's view). For one thing, it is advocated by existing design methodologies, which provide different bases for performing it. For another, the majority of classic empirical studies, especially —but not only— those conducted in software design, presented decomposition also as a very important, if not the main strategy in expert design activity.

A frequently observed source of design-activity misrepresentation that we have identified in early design studies (Visser, 1994a) is confounding the structure of the *result* of an activity with that of the *actual activity* (Carroll & Rosson, 1985; Visser, 1988a; Visser & Morais, 1988). This may be seen at work from Eastman (1970)'s famous study on. Working in the theoretical context of the SIP approach, Eastman notices in his conclusion that a "weakness" of his studies is that he "only gained a few protocols that clearly show hierarchical processes where one element is designed of smaller elements which in turn become part of a larger design. Surely, this is a central feature of many design problems". (1970, p. 32)
By definition, each design problem, in the sense of the global design project requirements, is "decomposed", in that it is transformed into several other problems to be solved. As used in the early design studies, however, the concept conveys particular presuppositions, such as that decomposition takes place in a systematic way, i.e. that it proceeds from "hierarchical planning" and that it implements "balanced" (especially top-down breadth-first) solution development (see Visser & Hoc, 1990, Carroll, 1985 497 for a critical presentation).
Even if it is only one amongst several possible approaches, decomposition of the problem to be solved is, indeed, generally used in order to plan the activity that is to solve the problem by means of solving its resulting sub-problems. However, often this planning is not hierarchical, this planning doesn't lead to a "balanced" problem decomposition, the consecutive solution development doesn't implement the resulting plan, which may indeed be questioned.

Example. The software engineer observed by Visser in her Software design study, decomposed his activity according to two lines. On the one hand, he divided his global industrial programmable controller (IPC) task into three consecutive parts according to the relative urgency with which various colleagues (especially in the workshop) needed these different parts. On the other hand, he decomposed the IPC software that was to be developed, into various modules corresponding to machine tool functions. His breakdown didn't stick to a hierarchical plan, but it followed the order in which modules appeared on the listing of an "example" software that the designer had previously developed for a similar machine tool installation and that he reused. This decomposition of the future software then guides the further planning of each of the three parts resulting from his task planning.
After his one-hour planning phase, the designer started to code. Further "planning" during coding was local. It took place at varying problem-solving levels and concerned more or less large entities (e.g., at the design level, it could concern a function or a machining operation; at the coding level, a module or an instruction). The engineer frequently deviated from the plan that he had formulated based on the example program. He did so if another organisation was judged more "economical" from the point of view of "cognitive cost". So the only decomposition, other than local planning, performed by the software engineer observed by Visser (1987) was guided by the urgency of components, and the existing decomposition of an example that the designer adopted for reuse.
Other authors have observed this absence of global decomposition and planning. Détienne (1994), e.g., notices that only at a micro level, one may observe a succession of planning processes and coding processes, constituting planning and coding mini-stages.
Such a scattered way of proceeding is, however, not due to designers being nonchalant or incompetent. "Disciplined" structured approaches such as proposed by waterfall models "seem to be unfeasible as a general technique, because the consequences of higher-level decisions cannot always be worked out fully until lower level ones are developed" (Green, 1990, p. 119).
One may frequently come across authors stating that the waterfall model has been replaced by an iterative approach to software development. At least two remarks may be formulated to this respect.
(i) Such replacement may have occurred in the conclusions of research papers and/or in descriptive contexts, but the waterfall model still has a powerful position as prescriptive reference underlying design methodologies —and software environments. It is, e.g., "widely used on large

government systems, particularly by the Department of Defence" (as noticed in http://asd-www.larc.nasa.gov/barkstrom/public/The_Standard_Waterfall_Model_For_Systems_Development.htm).
(ii) Presenting software development as an iterative process can only suffice as a first global characterisation. Indeed, in order to characterise the process, one needs to identify, at least, the conditions under which iterations occur, but also the components of the cycles iteratively traversed, which often will be rather low-level (see e.g. Malhotra et al., 1980).

Planning and organising design: the opportunistic organisation of design. Already back in 1980, Green (1980) in a paper on "Planning a program", concludes a discussion of structured programming methods by advancing the idea that "good programmers…. leap intuitively ahead, from stepping stone to stepping stone, following a vision of the final program; and then they solidify, check, and construct a proper path. That proper path from first move to last move, in the correct order, is the program, their equivalent of the formal proof". (Green, 1980, p. 306) Green notes that the author who introduced the concept of "stepwise refinement", Wirth, is himself "quite explicit; having described his stepwise refinement, [Wirth] says 'I should like to stress that we should not be led to infer that the actual program development proceeds in such a well organised, straightforward, top-down manner. Later refinement steps may show that earlier ones are inappropriate and must be reconsidered.'" (ibid.)
Since the eighties of the XXth century, more and more empirical design studies have shown that the actual organisation of design activity is not a "simple" implementation of a designer's plan. This conclusion corresponds to more general observations concerning design. Planning a complex activity such as many design tasks constitute, is a design activity in itself (see below the section on Planning as design). The remark that "design never ends" —neither with implementation, nor with maintenance, nor with actual use of the artefact— holds particularly for this design-activity planning.
The qualification that empirical studies of actual design activities have come to adopt for the way in which designers actually organise their activity is "opportunistic" (Visser, 1987, 1988a, 1994a; Visser & Morais, 1988). Inspired by Hayes-Roth and Hayes-Roth (1979), who have modelled errand planning as an opportunistically organised activity, we qualified the organisation of design activity as "opportunistic". We did so because designers do not follow systematically a pre-established plan —hierarchical or otherwise— but also use other resources to select successive design actions. This selection of design actions is determined by an evaluation function that is primarily based on the "cognitive economy" criterion. Pre-established plans may play a role, and often their role will be important, but they are only one of several possible resources that provide opportunities for action.
We have observed this opportunistic design in the different domains we have examined (see Table 1). In these studies, we identified six types of categories of data that could be "taken advantage of", as factors leading to the opportunistic organisation of design (Visser, 1990, 1994b):
1. Information provided to the designer by an external information source (in particular, the client or a colleague).
2. Information the designer "comes across" when "drifting" (i.e., involuntary attention switching to a design action other than the current one).
3. Information constructed by the designer when considering data from another viewpoint.
4. Information constructed by the designer as a by-product of problem-solving actions such as analysing the problem specifications, developing, and evaluating solutions, particularly, the evaluation consisting in exploring the implications and consequences of a proposed solution.
5. Mental representations of design objects activated by the representations used for the current design action, because of activation-guiding relationships existing between the two sets of representations.
6. Mental representations of design procedures that are striking because they have just been developed for the current design action.

NOTE: Guindon, Krasner, and Curtis (1987) speak of "serendipitous design", observing that problem solving is controlled by recognition of partial solutions, at different levels of abstraction or detail of the solution, without designers having previously decomposed the problem into subproblems. This serendipitous design may be occasioned by several of the above-mentioned opportunities.

Example. This example comes from our Functional specification study (for the other categories and examples, see Visser, 1994a). It is an example of designers taking advantage of mental representations of design objects connected by the relationship of analogy to the representations that they are using for their current design action (category 5).
The designer had planned to specify first-phase tooling operations (used in order to shape the rods) before second-phase tooling operations (used in order to finish the rods). The actual specification of these two types of operations is, however, completely intertwined. Considering second-phase operations as analogous to the first-phase operations, the designer, during his working on the first-phase tooling function, continually switches between first-phase and second-phase specifications. Often he takes advantage of the specification of a first-phase operation O1 in order to specify, by adaptation of this O1 specification, its corresponding second-phase operation O2. Frequently, an O2 specification "makes him think" of an omission or error made on the corresponding O1 operation.

The observation that designers organise their activity in an opportunistic way is not restricted to inexperienced designers. On the contrary, it is something typical of expert designers. Nor is it the translation of a deteriorated behaviour that occurs only when designers are confronted with a "difficult" design task. Even when expert designers are involved in routine tasks, the retrieval of pre-existing plans does not appropriately characterise the organisation of their actual activity. Through an analysis of 15 empirical studies of design, Visser (1994a) has shown that
even if designers possess a pre-existing solution plan for a design problem,
and if they can and, in fact, do retrieve this plan to solve their problem (which is often possible for expert designers confronted with routine design),
yet if other possibilities for action ("opportunities") are also perceived (which is often the case in real design)
and if the designers evaluate the cost of all possible actions ("cognitive" and other costs), as they will do in real design,
the action selected for execution will often be an action other than the one proposed by the plan: it will indeed be a selected opportunity.
Pre-existing plans that —if they are invariably followed— may lead to systematically organised activities, are only one of the various action-proposing knowledge structures used by designers. They may be interesting from a cognitive-economy viewpoint because executing an action for which such a schematic memory representation is available may cost relatively little if all schema variables relevant for execution have constant or default values. However, if other knowledge structures propose relatively more economical actions, designers may deviate from such a plan. This is especially true for experts, who may be supposed to possess —or else to be able to construct without difficulty— a representation of their activity that allows them to resume their plan later on, when it once again becomes profitable to do so from the viewpoint of cognitive economy. Having several possibilities for action to be compared and taking into account actions' cognitive cost, are two task characteristics which probably only appear in "real" design. This may explain why in laboratory experiments mostly systematically organised design activities were observed.
"Opportunities" must be "perceived": this perception is data-driven. It is on the basis of their knowledge that expert designers process the data that they perceive in their environment and that may take different forms: information (the state of design in progress —category 2 among the six opportunism-causing categories—, but also other information at the designers' disposal:

information they receive —category 1—, information they construct —categories 3 and 4), permanent knowledge and temporary design representations (in particular, the designers' representations of the state of design —categories 5 and 6). Taking advantage of these "opportunities" rather than following a pre-established plan will, indeed, lead to an opportunistically organised activity.

No independent stages in design problem solving

In the "Stage models and process models" section, we noted that problem solving rarely proceeds by traversing consecutively several independent stages. This is also one of the characteristics of an opportunistically organised activity: it is impossible to identify, in a design process, independent stages occurring consecutively, even if in iterative cycles.

The observation formulated in many empirical studies conducted on actual design activities, that solving design problems cannot be analysed as a linear process composed of several stages (e.g., the two consecutive stages proposed by Simon, i.e., problem structuring and problem solving) is also clearly related to the ill-defined character of design problems. Design problem solving is not a process going from analysing "the" problem specifications to synthesising "the" solution: problems do not pre-exist to solutions, both are built up and elaborated simultaneously. Simon's two-stage "componential" character of problem solving is another premise of the SIP model that, according to Mayer (1989), is being denied by the reality of ill-defined problems.

For authors such as Schön, this intertwined character of "problem structuring" and "problem solving" (in Simon's terms) is obvious. The author also links this observation to design problems' ill definedness. "Although the programme or brief to be satisfied by a design may be highly specified, design situations are always partly indeterminate. In order to formulate a design problem to be solved, the designer must *frame* a problematic design situation: set its boundaries, select particular things and relations for attention, and impose on the situation a coherence that guides subsequent moves. Moreover, the work of framing is seldom done in one burst at the beginning of a design process. Designing triggers awareness of new criteria for design: problem solving triggers problem setting". (Schön, 1988, p. 182)

The intermingled character of design and implementation

Not only design activities are intermingled, this also holds for design and implementation activities. For software design, this means that analysing actual software design —rather than coding— is not simple. Both types of activities are indeed often interwoven (Visser, 1987, 1992).

Example. The interweaving of design and coding was observed in our Software design study. We observed that the first day of work on this software was dedicated to an initial, global understanding of the problem. The software engineer "analysed", i.e., in fact skimmed through, the specifications he had received (a 50 cm pile of A4 documents). At the end of this first day, during one hour, he made a global plan for his coming activity on the IPC software. After this one day of problem analysis and planning, i.e. from the second day on, the software engineer directly started to code. This coding was —of course, we would say— often interrupted for one or the other of various other types of activities, from planning to solution evaluation, but not necessarily in iterative cycles of activities executed in a fixed order. Thus, the software engineer's interruptions of the coding activity were not systematic. That is why his activity during the rest of the four weeks that he took for developing the software, could not be further decomposed into functionally homogeneous phases. We concluded that design took place during the entire development process, completely intermingled with implementation of the resulting design choices (Visser, 1987) (cf. Carroll, 2000's "actively synthetic design method of planning by doing that is complementary to the relatively analytic techniques of problem structuring and decomposition", p. 29).

NOTE: Another interpretation of these observations would be that the software engineer completed his entire design in only one hour, that is, that there wasn't nearly any design taking place —the software engineer's activity consisting in directly, or almost exclusively, coding the concept he designed in one hour.

Activities related to opportunism: constraint posting and requirements engineering
The principle of constraint posting, especially the motives underlying this activity, can be linked to the opportunistic character of problem solving. "Opportunistic solving refers to taking the opportunity of solving one subproblem while the solver is in the process of solving another subproblem. Similarly, constraint posting refers to not solving a subproblem until it becomes necessary to deal with a constraint during the course of solving the subproblem". (Voss et al., 1983, p. 168)
A refutation of the viewpoint on requirements engineering according to which requirements are "out there", and can be gathered once and for all at the beginning of a design project, is well illustrated in a study by Carroll, Rosson, Chin and Koenemann (1997). In the course of a series of participatory design activities, the authors observed how the nature of the project requirements evolved. Requirements evolving during the design process is not a matter of "initially mistaken notions being subsequently corrected, or of more requirements work leading to successively finer decompositions" (p. 62). The authors analyse the process of requirements gathering as requirements "development" (see also Carroll, 2000).

Decomposition and planning *Planning and organisation.* As conclusion of the two preceding sections, let us resume their combined result. A designer's problem decomposition leading to an action plan, does not mean that the designer is going to observe this plan. The actual organisation of design activity will generally be opportunistic, following only in part the plans that were developed!

euse vs. design "from scratch". All use of knowledge could be called "reuse" in that knowledge is based on the processing of previous experience and data encountered in the past. We reserve "reuse" (vs. other "use" of knowledge) for the use of specific knowledge that is at the same abstraction level as the "target" for whose processing the knowledge in question is retrieved (Visser, 1995a).

Terminology
In the research literature on analogical reasoning and reuse, "target" is the term adopted for the problem to be solved (or, in other than problem solving tasks, the material to be processed). "Source" is the term adopted for the knowledge used in order to solve the target problem: in analogical reasoning and reuse, it is generally a solution to another, analogous problem.
Thus, "reuse" of knowledge is opposed to the use of more general, abstract knowledge (such as knowledge structures like schemas and rules).

Reuse has been identified in various empirical design studies. The exploitation of specific experiences from the past is indeed particularly useful in design, especially in non-routine design (Visser, 1995a, 1995b, 1996; Visser & Trousse, 1993)
Before cognitive psychology and cognitive ergonomics started to examine reuse, researchers in computer science had already identified and examined it as a research topic. Authors in the software engineering community had defined the domain of "*software reuse*" (Biggerstaff & Perlis, 1989a, 1989b)[41].

[41] At the conference considered to be the birthplace of "software engineering", i.e. the 1968 "NATO Software Engineering Conference", the question of software reuse was initiated, right from the start, by an

In Artificial Intelligence, a phenomenon similar to reuse has been analysed in terms of "case-based reasoning" (CBR). The CBR community also has argued that this form of reasoning is particularly well suited to design problem solving (see e.g. Pu, 1993) (see Visser, 1994d, 1993c; Visser & Trousse, 1993).

Software-engineering researchers distinguish "design for reuse" from "reuse for design" (Thunem & Sindre, 1992). The construction of reusable entities that are to be organised into a "components library" is often considered an independent design task, not necessarily executed by the designer who is going to reuse these entities. We are unaware, however, of any empirical study conducted in such "design for reuse" situations. Existing empirical studies concern "reuse for design" and show that the two activities are not as separate as software-engineering researchers suppose.

A considerable proportion of the empirical, cognitive ergonomics research on reuse in design has been conducted in the domain of software, especially that of object-oriented (OO) software (Détienne, 2002a). Other paradigms have also been examined. Visser (1987) observed reuse in IPC-software design using a declarative type of Boolean language. Weber (1991) has examined reuse in LISP. Visser has also studied reuse in other domains of design, i.e. mechanical (Visser, 1991b) and industrial design (Visser, 1995b).[42]

Several aspects of reuse have been examined in these studies. Here, we focus on the question that is at the basis of reuse-based design, i.e.: When do designers decide to adopt reuse in order to solve a design problem, rather than base their problem solving on general knowledge, i.e. proceed to "design from scratch"? For other aspects, such as phases of reuse, strategies of reuse, types of entity reused, types of exploitation of reusable entities, effects of reuse on designers' productivity, and difficulties and risks of reuse, see, e.g., (Détienne, 2002a) or Visser (1995a).

Reuse takes place in, at least, five phases:

0. construction of a representation of the target problem;
1. retrieval of one or more sources;
2. adaptation of the source into a target-solution proposal;
3. evaluation of the target-solution proposal;
4. integration into memory of the resulting modifications in problem and solution representations.

NOTE. This five-phases model is a particularisation of the general three-phases problem-solving model —with a memory-integration phase appended[43]):

1. construction of a problem representation (reuse phase 0);
2. solution generation (reuse phases 1 and 2);
3. solution evaluation (reuse phase 3).

This comparison shows the global character of the three-phases problem-solving model, and/or the attention paid in research on reuse to the solution-generation mode particular to reuse. This is of course the solution-generation mode used in analogical reasoning. For differences between analogical reasoning, reuse, and CBR, see Visser (1999b).

The construction of a target problem representation (phase 0) has seldom received any attention in empirical studies, neither in studies on analogical reasoning, nor in those on reuse. It is, however, during this phase that designers have to decide whether they are going to adopt —that is, try to adopt— reuse in order to solve their design problem. We are not aware of any study presenting empirical data on designers taking into consideration the choice between design from

invited paper delivered by McIllroy, entitled "Mass Produced Software Components" (Krueger, 1989). Since these days, many research has been conducted on the reuse of software, especially in the domain of object-oriented design, paradigm of which reuse is one of the basic principles (see e.g. Gamma, Helm, Johnson, & Vlissides, 1995a, 1995b).

[42] We have also analysed reuse in diagnostic activities (Visser & Perron, 1996a, 1996b)

[43] The fact that such a phase normally is not being attended to in problem solving research, seems to us a consequence of the traditional separation of "functional departments" in psychology (especially, memory, learning, language, problem solving).

scratch or design based on reuse —rather than to "simply" design from scratch right from the start. Two individual studies seem to indicate that the "cost" of reuse is an important, if not the main factor in this decision process (Burkhardt, 1997; Visser, 1987).
In an experimental study, Burkhardt asked seven OO-software designers to describe elements they might want to (re)use. Half of his participants mention that there are reus*able* elements whose actual reuse they would not envision because of the "cost" of their reuse.
Data gathered by Visser (1987) present an example of a factor contributing to the cost of reuse —but only once candidate sources, i.e. reusable solutions, have been retrieved. This factor is the cost of required adaptation, itself a function of target-source similarity. Sources are indeed always to be adapted in order to be usable as a possible solution to a target problem that is "similar" to the source problem that had been solved by the source[44]. This conclusion coincides with a position adopted in several CBR systems in which the selection of a "case", i.e. a reusable source, is guided by its adaptability (Keane, 1994; Lieber & Napoli, 1997; Smyth & Keane, 1995).
The importance of the "cost" factor in the choice of a strategy such as reuse is completely in line with our identification of the primordial factor underlying the organisation of design, i.e. cognitive economy (Visser, 1994a). If designers are conscious of several action possibilities, the relative cost of an action determines their choice.
With respect to the frequency of reuse actually taking place, different authors in the domain of software engineering assert that 40 - 80 % of code is non-specific, thus reus*able*. Many authors advance the percentage of 80 %, mostly referring to T. C. Jones (1984), who summarises four studies conducted between 1977 and 1983. In 1989, however, Biggerstaff and Perlis assert that actually, "over the broad span of systems, reuse is exploited today but to a very limited extent".
Empirical studies providing quantitative data about actual reuse all concern OO software, which, due to its mechanisms of inheritance, abstraction and encapsulation, and polymorphism, is considered to particularly "favour reuse". The conclusions of these studies may thus be specific to this particular software design paradigm. As far as we know, the only empirical study observing "massive" reuse concerns OO software (Lange & Moher, 1989).
If empirical studies of other design tasks do not provide data on the frequency of reuse, experimental studies of analogical reasoning can inform us. Their general conclusion is that source retrieval seldom occurs "spontaneously", i.e. without experimenter suggestion, which indeed often occurs in these laboratory studies (the Lange and Moher 1989 study being an exception). In professional work situations, however, one cannot expect that external persons hint designers to reuse —except for general reuse encouraging enterprise policies.
The data available on reuse-based design vs. design from scratch is thus still very meagre. Nevertheless, it is a central topic with respect to reuse, its role in design, and the possibility to support designers in their reuse during design.

Reuse, creativity, and novelty

In the section discussing the role that Simon attributes to decomposition, we remarked that this method generally leads to standard, routine solutions because of its top-down nature. This remark should not lead, however, to belief that bottom-up or data-driven problem solving is a guarantee of novelty, whereas a top-down or concept-driven way of proceeding necessarily leads to routine solutions.
Example. In an analysis of two different types of analogy use (Visser, 1996), one of our conclusions was that according to the type of use made of analogy, analogical reasoning may add to the innovative nature of the problem-solving activity and/or its result, or may not. Augmentation of creativity generally seems to occur in analogical reasoning at the action-execution level (see the

[44] Adaptation by "reasoning from precedents" (past experiences) is one of the two aspects of innovative design on which Navinchandra (1991) focuses (exploration being the other one).

section on Problem-representation construction, solution generation, and solution evaluation), leading to the type of analogy use on which psychological studies have focused. The second eventuality occurs with analogical reasoning at the action-management level. Especially on the level of structure, the transfer of previously applied solutions into a new design project may introduce homogeneity into that project and, because of that, remove a possibility of novelty in the resulting design!
In their study of opportunistic planning, Hayes-Roth and Hayes-Roth (1979) observe that "the bottom-up component in multi-directional processing provides a potentially important source of innovation in planning" (p. 306). The authors also refer to Feitelson and Stefik (1977) who made similar remarks concerning the "largely event driven nature of the planning process" used by an expert geneticist whom they observed. The expert is supposed to proceed in this event-driven way because of his "fishing for interesting possibilities" (ibid.). Our previous observations lead us to emphasise that data-driven problem solving indeed constitutes a "*potentially* important source of innovation", but that one should not conclude that it is always, or by definition, leading to more innovation than a concept-driven way of proceeding!

Simulation. Simulation is often presented as an important evaluation strategy. It is, nevertheless, also used —and useful— in generation.
Mental simulation may be used "in order to identify relevant knowledge about the problem. It is a means of accessing knowledge from the designer's [long-term memory]". (Baykan, 1996, p. 141)
For Adelson, Littman, Ehrlich, Black, and Soloway (1985), mental simulation of the design is the main operator used to check the sufficiency and the consistency of the current design state, and to expand the design from one level of specificity to the next.
Even if simulation is often mental, in mechanical and industrial engineering, designers also use physical devices. In a comparative analysis of "manual" design and computer-assisted design, Dillon and Sweeney observed that "movement of parts of the design was often simulated on the board by using cut-outs of the relevant shape and literally moving it around the drawing by hand. This acted as a powerful visual aid to some designers. No directly comparable facility existed on CAD". (1988, p. 484)
Guindon, Krasner, and Curtis (1987) underline the importance of understanding and elaborating the requirements through mental simulations, which they specify as exploration of the designer's mental model of the problem environment.

User considerations guiding development and evaluation. In user-centred system design, knowledge about the users of a system may guide both system development and system evaluation (Norman & Draper, 1986). Traditionally, the reference adopted by system designers for their specification of interactions between user and system is models of user tasks rather than of user activities. System designers indeed often adopt task analysis models and techniques, such as GOMS (Card, Moran, & Newel, 1983).
Task analysis has undergone critical analysis. It has been argued, e.g., that task analysis "(if used on its own) would produce poor system designs because it fails to achieve sufficient device independence" (Benyon, 1992). Task analysis has also been qualified as "pseudo-cognitive". It indeed serves to identify and analyse users' tasks rather than their activities (see the section on Task vs. Activity). Functional specifications for interactive systems may require, however, data on actual activities.
More recently, scenario-based design (Carroll, 1995, 2000) and participatory design in general (Kyng & Greenbaum, 1991) are approaches used to gain data on actual users' activities, aiming to take into account such data in systems to be designed.
Software system usability testing and more general evaluation methods of interactive systems can proceed by laboratory user testing or apply more rapid techniques, referring to guidelines or

other lists of inspection points (such as cognitive walkthroughs, heuristic evaluations, or even rapid ethnography, Millen, 2000).
As far as we know, little empirical research has been conducted on the way in which a designer's activity is guided by user considerations. Our Software design study has disclosed some aspects of this strategy.
Example. The observed software engineer designing IPC software tried to take into account several types of users who were going to work with the IPC that he was designing (Visser, 1987). He judged the system operators in the workshop as its most frequent users. Maintenance personnel were considered another category of IPC users. The engineer was observed to take into consideration these two types of users. Considering homogeneity an important factor of ease of use, he used it as a design constraint, trying to make the software as homogeneous as possible, both for comprehension by the system operators, and for maintenance reasons. This search for homogeneity was realised in various ways, using the homogeneity constraint both in development and in evaluation of the software.
Using it as a generative constraint, the software engineer created uniform structures at several levels of the software: he homogenised the instruction order in the modules, the branch order in the instructions, and the bit order in the instruction branches.
Evaluating his software against the criterion of homogeneity sometimes led the engineer to reconsider instructions. In doing so, he modified either the current instruction in making it replicate the structure of previously written ones; or previously written instructions by giving them the structure of the current instruction.

As early as 1972, Rittel (Interviewed by Grant and Protzen, 1972/1984), in a discussion of planning societal problems (governmental planning, especially social or policy planning), states that "there is no professional expertise that is concentrated in the expert [designer]'s mind, and… the expertise used or needed, or the knowledge needed, in doing a design problem for others is distributed among many people, in particular those who are likely to become affected by the solution —by the plan— and therefore one should look for methods that help to activate their expertise. Because this expertise is frequently controversial, and because of what can be called 'the symmetry of ignorance' … the process should be organized as an argument". (p. 320) Rittel's expression "symmetry of ignorance" refers to his conviction that there is no reason (logical or educational) to defend that one "expert", i.e. one particular participant in the whole decision process, knows better than another. Rittel considers this "symmetry of ignorance" a "non-sentimental" argument for design participation.

4. Design of plans

"Planning" globally refers to two activities, plan elaboration —i.e. design of plans— and plan execution —i.e. implementation of plans. Of course, these two types of activities aren't independent. Planning in the sense of plan execution often involves replanning —that is, not only modifying existing plan components, but also elaborating new plan components. On the other hand, as holds for all design activities, design of a "new" plan will rarely proceed from scratch (Visser, 1994a). In the present context of design, discussion only concerns plan elaboration.
NOTE: In the KADS knowledge elicitation domain (see Breuker et al., 1987), design and planning are considered to differ with respect to their object: design concerns artefacts, planning temporal events.
Even if planning —and organising one's activity— might be considered design activities, they have rarely been analysed in these terms before we did so, in the framework of research concerning the way in which designers organise their activity. Besides, most research on planning has been concerned with execution rather than elaboration of plans (Hoc, 1988). The few studies that have been conducted on plan elaboration are now some 25 years old (Byrne, 1977; Hayes-Roth & Hayes-Roth, 1979). In our research on designers' organising their activity, we have conducted several studies identifying different types of plan deviation, and analysed the factors un-

derlying these deviations in terms of an opportunistic model (Visser, 1994a) (this research has been presented above).
Planning in the sense of plan elaboration, is a design task that, in several respects, differs from the design tasks generally analysed in cognitive design research. Some of the contrasting dimensions are: the professional - everyday character of the design task; its individual - collective character; and the type of constraints involved, especially the role of temporal constraints.
Compared to other design tasks, planning is often performed in non-professional, everyday contexts. Contrary to professional design tasks, it is mainly conducted individually. Both early plan design studies on route planning (Hayes-Roth & Hayes-Roth, 1979) and meal planning (Byrne, 1977) examined such types of planning. It was the famous study by Hayes-Roth and Hayes-Roth (1979) on route plan design that has led to the development of the opportunistic model for planning, which since has been widely adopted for modelling other types of design tasks —even if Hayes-Roth and Hayes-Roth themselves did not formulate a link to the activity of design in general.
With respect to the types of constraints involved, planning, by definition, is essentially a question of temporal constraints, as noted above in the section on "Constraints on design solutions". The specifications defining the items to be organised into a plan may be open with respect to the items' temporal characteristics. However, the planning activity introduces such constraints: items to be organised into a plan are to be organised on a time axis. This holds for all types of planning, whether the artefacts have spatial characteristics (as in route planning) or not (as in meal planning).
In addition to the specificity of plan design studies just mentioned (their typically non-professional and individual character), route planning is often, the affair not only of lay people, but even of "novices", if not in the type of design per se (any driver may recurrently proceed to route planning), at least with respect to knowledge of the "application domain", i.e. the environment to be traversed. It is exactly when they are planning to drive, or actually driving, through an unfamiliar region, that people are forced to plan their routes.

4. Design of software

In his Introduction to "Bringing design to software", Winograd (1996), under the heading "What is design?", qualifies design —that is, software design— in several sections entitled:
Design is conscious
Design keeps human concerns in the center
Design is a conversation with materials
Design is creative
Design is communication
Design has social consequences
Design is a social activity.
These different qualities of software design are to be discussed in the various contributions to the book. Winograd concludes his Introduction asserting that "what is common to all the authors" is "a concern for the situated nature of design —a sensitivity to the human context in all its richness and variety". (1996, p. xxv) The aspects of design that we discussed above (the various characteristics of design problems and of design activity) do not appear in any of the chapters. They thus add elements to this list of aspects that may contribute to elucidate the "human context [of design] in all its richness and variety"….

4.1 Empirical cognitive studies of software "design"

Even if the first empirical design studies examined architectural design, there have also been various early empirical studies of software design, especially by Green (1980), and Carroll and colleagues (Friedman, Malhotra, Miller, Thomas and Rosson) (Carroll, Thomas, & Malhotra, 1980; Carroll, Thomas, Miller et al., 1980; Malhotra et al., 1980; Thomas & Carroll,

1979/1984). In addition, software design has probably been the domain in which the biggest number of empirical, cognitively oriented studies has been conducted.
Several paradigms have been examined. The first studies of software design (or rather on "programming" in the sense of coding —see our remarks hereafter) concerned procedural languages such as Fortran or Pascal (Brooks, 1977). More recently, studies have been examining the object-oriented paradigm, through studies of C++ (Détienne, 2002a). As noticed above with respect to reuse in software design, a considerable proportion of the empirical, cognitive research on programming and/or software design has been conducted in the domain of OO software; LISP and a declarative type of Boolean language have also been studied.
For at least two reasons, several of these studies do not confer much knowledge concerning "design". First, many of them concern programming in the sense of coding, i.e. implementation of design decisions, rather than design, i.e. the elaboration of these decisions. Even if in several of these studies of program coding, the participants also had design tasks to fulfil, few authors have analysed the corresponding design activities. This may explain that the research domain that was created was called, in its early years, "Psychology of programming" and that the workshop that has come into life in these years was entitled "Empirical Studies of Programmers" (ESP) (Cook, Scholtz, & Spohrer, 1993; *Empirical Studies of Programmers (ESP) Seventh orkshop*, 1997; *Empirical Studies of Programmers Eighth orkshop*, 1999; Gray & Boehm-Davis, 1996; Koenemann-Belliveau, Moher, & Robertson, 1991; Olson, Sheppard, & Soloway, 1987; Soloway & Iyengar, 1986; Wiedenbeck & Scholtz, 1997) (see also Détienne, 1998; Gilmore, Winder, & Détienne, 1994; Hoc, Green, Samur ay, & Gilmore, 1990). Second, most, if not all of these studies concern small tasks —be it design or coding— whereas design is an activity that has many characteristics that only appear in "real" design tasks (Visser, 1987).
Our longitudinal study concerning three stages in the design of industrial programmable-controller (IPC) software, conducted in the years 1987-1988 (see Table 1), was among the early "programming in the large" research —as opposed to the "programming in the small" research that characterised most studies conducted in those years. One of the questions raised at the end of the First ESP workshop (Soloway & Iyengar, 1986), and which subsequently became the title of a Future Directions paper, was "By the way, did anyone study any real programmers?" (Curtis, 1986). In our IPC Software design study (Visser, 1987), we indeed analysed the design activity of a "real programmer" and corroborated the hypothesis at the source of this study, i.e., that strategies used in a professional design setting differed, at least partially, from those that had been observed until that day in studies conducted on most novice, student programmers working on artificially limited problems.
All design characteristics presented above hold for software design. The design strategies discussed have all (also) been examined in studies of software design. We will only come back on these aspects, when they have specificity in situations of software design.

4. 2 "Software design" as a separate profession in the software domain

We defended above the view introduced by Simon according to which not only so-called "designers" are involved in activities that, from a cognitive viewpoint, are appropriately qualified as "design". We mentioned that the idea that the term "designer" is being used inappropriately has also been defended from other than cognitive viewpoints.
Many software designers themselves have still another viewpoint on this question. They want "Software Designer" to be an independent profession, recognised through a job title. They are concerned by the cleavage between their professional status as "software engineers" or "programmers", and their executing an activity that they consider "design" —even without adopting a cognitive viewpoint.
In January 1991, the 15th anniversary issue of Dr. Dobb's Journal of Software Tools (DDJ) focused on the design of software, with, amongst other papers, a manifesto by Kapor (reprinted in Kapor, 1996; Winograd, 1996). This DDJ issue, especially Kapor's contribution, advocated that software design ought to be a separate discipline in the software domain. Kapor defends the

specific role of design, drawing a parallel between fabricating software and fabricating buildings. According to the author, what we have "nowadays" (i.e., in 1991) are construction workers designing software whereas we need software architects. Such software designers need formal training and recognition as members of a separate profession, equal to computer scientists and engineers.
Five years later (1996), in his DDJ column "Design: Whose Job Is It, Anyway?", Swaine (Retrieved 27 August 2003, from http://www.ddj.com/print/documentID=13370) writes that over the last ten years, in some organisations, "Software Designer" has become "a real job title", and that "there are schools offering multicourse programs in software design". Swaine notes that "books on software design aren't as common as books on Java programming", but there are several, "excellent" ones[45].
In 1992, a new professional organisation was founded, the Association for Software Design (ASD). The ASD "delights in revealing to prospective members that they have been engaged in software design, even though their payroll records may refer to them as software engineers, as programmers, as program managers, as human-factors consultants, or as one of many other titles". (Winograd, 1996, p. xv)

4.□3 Models of software design: the waterfall model as a lasting reference

As in other domains of design, many models of software design have been proposed without any reference to the actual activity. They are often used as the basis for managing software development. So is the Standard Waterfall Model for Systems Development, a stage model, often abridged called "the waterfall model" (Boehm, 1981, as quoted in many papers on design, e.g. Kitchenham, 1990 □164, see also http://asd-www.larc.nasa.gov/barkstrom/public/The□Standard□Waterfall□Model□For□Systems□Development.htm). This widely used model defines a certain number of stages that the software development process is supposed to go through. Design, e.g., is supposed to take place before coding, and requirements are to be defined before one starts to design.
As also holds for the stage models in other domains of design, the model does not reflect, however, the actual activity of people involved in software development. This is not only a conclusion achieved at by psychologists on the basis of empirical studies (Détienne, 2002a; Gilmore et al., 1994; Hoc et al., 1990; Visser, 1992) (see also the discussion in the "Decomposition in design" section), software engineers themselves also formulate this judgement (Kitchenham & Carn, 1990; Löwgren, 1995).
Some specific criticisms of the Waterfall model, not especially of cognitive nature, are the following (http://asd-www.larc.nasa.gov/barkstrom/public/The□Standard□Waterfall□Model□For□Systems□Development.htm):

- The model can be very expensive.
- Requirements are supposed to be fixed before the system is designed, but requirements generally evolve.
- Designing and coding often turn up requirements inconsistencies, missing system components, and unexpected development needs.
- Many problems are not discovered until system testing.
- System performance cannot be tested until the system is almost coded; problems such as undercapacity may then be difficult to correct.

The Waterfall model is also associated with the failure or cancellation of a number of large systems. As a result, the software development community has experimented with a number of alternative approaches, including Spiral Design (also proposed by Boehm, 1986, quoted in

[45] Examples presented are "Tog on Interface" and "Tog on Software Design" by Bruce Tognazzini; "Computers as Theater" by Brenda Laurel; "The Essentials of User Interface Design" by Alan Cooper; "Things that Make Us Smart" by Don Norman.

http://asd-www.larc.nasa.gov/barkstrom/public/The_Standard_Waterfall_Model_For_Systems_Development.htm), Modified Waterfalls, (Evolutionary) Prototyping, Staged Delivery (see McConnell, 1996, quoted in ibid.). Commercial software projects often reduce the formality of the full Waterfall model. In the last few years, a paradigm known as Extreme Programming has emerged that emphasises reducing the cost of software changes, developing test cases before coding, developing code using pairs of programmers, and putting most of the documentation into the code (see Beck, 2000, quoted in ibid.). All these proposals have, however, not yet been evaluated from a cognitive viewpoint —even if they are becoming the object of empirical software engineering studies (Jedlitschka & Ciolkowski, 2003b).

4.4 Methods for software design: mathematical methods and structured methods

Methods proposed for software design are mainly of two types, mathematical methods, and structured methods. These different types of methods have been developed and are being used mainly by two different communities. Because of their complementary nature on several dimensions, integration of both is considered particularly useful by some authors, but has not yet been undertaken (Kitchenham & Carn, 1990).

Mathematical methods provide an unambiguous notation, but they offer no rules or heuristics in software development, e.g. for requirements formulation. They can be used for verification of software, but of course, only once this software has been developed.

Structured software design methods are manifold, as in other design domains (cf., e.g., the ASE-paradigm proposed by Jones, 1984, Originally published in 1963). Examples of families of structured methods proposed for software design are data-flow analysis and design (Yourden & Constantine, 1978), data-structure methods (especially, Jackson Structured Programming, JSP, applied in Jackson System Development, JSD, see Jackson, 1975; 1983, but also SSADM, Structured Systems Analysis and Design Method, a standard for British database projects), Unified Process, and, particularly in the domain of database design, entity-relationship and OO modelling. Contrary to the mathematical ones, these methods can be used for the specification of system structure. In many, if not most, commercial settings, people involved in design are obliged to use such methods with the aim of steering the process. In their actual activities, however, designers do not manage to conform systematically to such methods.

Methods especially advocated for the design of HCI seem to focus mostly on specific, even if central, aspects of interface design. Both classical task analysis and the more recently proposed participatory design approach focus on taking into account the user of the system under design, through gathering information on this user, in order to the guarantee user-adapted character of the system (but see our remarks formulated above concerning user considerations).

Various notations are also proposed for modelling interactive systems, but they are not integrated in terms of procedural practices that might provide guidance on their use (Budgen, 2002, oral communication).

Authors analysing software design from a cognitive perspective advance that design can only succeed, if designers are "allowed" to follow an iterative process of refinement and redesign (not necessarily corresponding to what is called "rapid prototyping", which may constitute a more "sketchy" approach to design), or an approach to design that Carroll (2000, p. 42) qualified as "cumulative", and in which an artefact can evolve through different forms all over time (cf. also Simon's approach to social planning) (see, however, our remarks formulated above concerning iteratively conducted cycles of activities).

4.5 Design of CI

Important aspects of software design, especially of interactive software design, concern what is commonly referred to as "human-computer interaction" (HCI), a field with clearly interdisciplinary concerns, but in which software design and ergonomics (and/or "human factors") probably

play the main roles. The domain of HCI indeed covers work on "interactive *computing systems* for *human use*" (as ACM SIGCHI defines the domain it is focusing on; italics added).
When one examines the references to research conducted in the domain of HCI, or more generally, of interactive software systems, many studies seem to relate to "design". A closer inspection shows that among the aspects of design that are being examined, few if no attention is paid, however, to the dynamic aspects of design as covered here. Most papers analysed and/or list different types of knowledge that are considered relevant for designers of HCI, i.e., knowledge that these designers need and are supposed to use in their activity of designing HCI. However, *the way in which* designers are to *use* this knowledge —and even less *the way in which* they are *actually* using it— is rarely being examined or discussed. One may encounter qualifications such as: "the HCI design model emphasises user-centredness during the design process", sustained by references to e.g. Norman (1986). However, neither Norman, nor the authors of these studies describe *how* it is that designers of HCI systems may *actualise* this user-centredness in their design activity.

□*he specificity of HCI design*. Design in the domain of HCI has been the object of much discussion for some 20 years now. Some authors in the domain note the similarities between design of interfaces and that of other types of artefacts. "In both the buildings and the user interfaces, a good design relies on principles such as ease-of-use and providing functionality that meets real needs.… Each design discipline, whether architecture, HCI, or automotive design, succeeds by meeting the functional needs of the user in an aesthetically pleasing manner". (Ford & Marchak, January 1997)
Many authors, however, insist on the specificity of user interface design, opposing e.g. software design and/or HCI design to engineering design. According to Winograd (1996), it is due to its user-oriented character that software design is contrary to engineering design, but comparable to architectural and graphic design. Winograd (1997) considers that the new field of "interaction design", leading to the "new profession" of "interaction designer", has computers at its centre, but is not a subfield of computer science.
Still other authors contrast "interaction design" with "interface design". Quoting Theodor Holm Nelson, Marion (Retrieved September 26, 2002, from http://www.chesco.com/□cmarion/Academic/AcFound.html) asserts that "learning to program has no more to do with designing interactive software than learning to touch-type has to do with writing poetry". "Interaction design answers the question 'how should this product work?' It tells us how the elements of the product work together in order to both make its functioning clear and enable the user to undertake her most important tasks easily. Interface design answers the question 'how should this product present itself?' It tells us how the product should look in order to maximize readability for the user, and includes the aesthetics of the product". (Retrieved September 26, 2002, from http://www.chesco.com/□cmarion/Academic/AcFound.html)
Many authors in the domain of HCI design declare that the design of interactive software is a completely different affair from design of other software. Different types of arguments are advanced, but not based on cognitive analyses of the underlying activity.
Among the papers that seem to tackle the topic, two types of approaches may be distinguished. There are authors who, in spite of their announcing a discussion of software design —be it interactive or not—, discuss related, but different themes (this is not specific to discussions of software design). There are also authors —sometimes the same as the previous ones— who seem to "discover" the topic of design and introduce their ideas as such. Presenting some observations, evoking one or more questions, they advance some general ideas without any reference to the numerous results and models in the domain of cognitive design research. This type of approach to design studies may be found, e.g., in the CHI community. Both empirical work and modelling approaches of the cognitive aspects of design activity are rare among contributions to the yearly CHI conferences. In 1995, "the CHI'95 conference instituted a new section called design briefings for presentation of notable designs and for discussion of how those designs

came to be" (Winograd, 1996, p. xiv). These contributions did not discuss, however, the cognitive aspects of this "coming to be".
Another example is a nine-page "perspective" in the SIGCHI bulletin "interactions" (Dykstra-Erickson, Mackay, & Arnowitz, 2001). In the form of "a trialogue on design (of)", the three authors present their contribution as disclosing a new, unexplored field —which it may be in SIGCHI: "this article should be looked upon as a starting point for [the] discussion [in SIGCHI]" (p. 109). Seemingly unaware of the existing tradition in cognitive design research, the authors develop as the "fundamental topic" in their paper, the question "what is 'design', and how do we define 'designers'?" (p. 109). In order to answer these questions, they introduce three different "perspectives on design" (presented in Winograd, 1996): Norman's culturo-organisational view, Sarah Kuhn's sociotechnical approach, and Kelley's idea that "design" and "designer" have a much broader coverage than generally acknowledged and that design, amongst other aspects, comprises "visualizing". All three perspectives have their value for the authors, but "design, in a cognitive sense…. is not necessarily visual at all", nor is it exhaustively characterised by the other perspectives. "A dilemma for the CHI community [is that] 'design' really isn't something that can be narrowly defined". (p. 113) And the authors to "propose" that design "needs to be further qualified" (ibid.). "There is room at the CHI table… to seat a good many practicioners, researchers, and engineers who all 'do' design of, in one way or another. And it is time to encourage the conversation to develop between all designers of…. Hence the continuing development of 'design'-oriented venues at CHI, and our request to you to contribute your ideas, submit to these design venues, and volunteer to be a part of the conversation". (ibid.)

4.1 Conclusion

This main section of the text has presented our approach to design. As this approach will also be discussed in the general Conclusion hereafter, we will restrict ourselves to some specific points on which we will not come back any more.
It seems significant to us that cognitive design research has concerned the activity rather than the product. Researchers have noticed that this product (the artefact that results from design activity) may take different forms without one being "better" than the other (see our section on Several satisfactory, rather than unique, correct solutions). Rare are, however, the studies analysing the quality of these products. In engineering, e.g. empirical software engineering (Jedlitschka & Ciolkowski, 2003a), researchers are concerned with measuring, or estimating, both the effort put into the process and the quality of the product. We have defended elsewhere the idea that measuring process effort and product quality and establishing a relation between the two cannot be performed without a model of cognitive activities involved in software design, and without measurement of these activities (Détienne et al., 2003). Today, the available data in these cognitive models with respect to this question is, however, still sporadic. In the section Quality of design problem solutions, we referred to one of the "German empirical studies" on engineering design (Fricke, 1999), which examined the link between cognitive design process variables and the quality of the resulting product. The fact that this study was conducted in the framework of a collaboration project with engineering design researchers, is probably no coincidence. The study identified several characteristics of the activities of "successful" designers. However, empirical design studies show that designers often do not adopt the procedures or strategies that correspond to these qualities of successful activity.
We noticed that empirical cognitive research on software design occupies historically a strong position in the research on design. The studies on cognitive aspects of HCI design have remained, however, rather limited with respect to the focus of the present text. They indeed focus on knowledge designers possess, and may or "should" use, rather than the dynamic aspects of this use (e.g., conditions of use, strategies). We do not suggest that the research in this domain is not concerned with relevant cognitive aspects. The impact of the artefact and the consequences

for user involvement in design are, e.g., topics that, on the one hand, are particularly central in the domain of HCI and, on the other hand, have not been much examined elsewhere.

5 Conclusion

In this conclusion, we will review our approach to design and we will discuss two types of questions. Concerning this last point, we will come back upon several themes that have been central in this text: SIP (symbolic information processing), SIT (situativity) and design, and design as problem solving. We will also discuss a question that has been touched upon in the design literature, but that —as far as we know— has never been the object of specific analysis, i.e. the influence of the nature of the design object (artefact) on the design activity.

As noted already, we are convinced that the expectations one has (one's theoretical framework or other attitudes) guide one's data collection —because they orient one's noticing and interpreting phenomena (see our remark concerning our overlooking reuse in our early software design studies). We also discussed another, frequently observed source of activities being misrepresented, i.e. confounding the structure of the *result* of an activity with that of the *actual activity*.

5.1 SIP SIT and design

It was by reference to Simon's idea of limited capacities leading to bounded rationality that Schön and Wiggins qualified designing as possessing "the conversational structure of seeing-moving-seeing" (1992, Spring, p. 143). It may be interesting to note that Simon himself did not realise this view in his approach to design. As observed already, Simon did not apply to his general model of design the qualified approach he adopted for social planning —and human economic behaviour. He discussed the bounded rationality idea, which was so central in his approach to economics (Simon, 1955), as relevant to his view of social planning. However, he did not do so in his analysis of design in general, which he presented through discussions of two other design domains, i.e. engineering and architectural design.

As noted several times in this text, our critique concerning the SIP approach to design concerns its too systematic, and thereby impoverished, approach to design. Simon indeed represents design, i.e. engineering and architectural design, as much more orderly than observed in studies on actual design activities —by the way, he never refers to such work. Simon's view applies to "simple", well-defined problems and to their processing, but does not represent the ill-defined problems that professional designers have to solve. This position has been substantiated through the discussion of six aspects on which the SIP approach tends to misrepresent design. Even if a model, by definition, simplifies its object, Simon's representation of design, in all its simplicity, no longer renders the specificities of design.

Greeno, once collaborating with Simon in the SIP paradigm (Greeno & Simon, 1988), came to adopt a situativity perspective. Among the proponents of this family of approaches, we consider him one of the most meticulous authors. We wish to recall the terms in which he presented SIP and SIT in the *Educational esearcher* debate on Situated Cognition/Action (January/February 1997). Greeno asserted that "the cognitive perspective takes the theory of individual cognition as its basis and builds toward a broader theory by incrementally developing analyses of additional components that are considered as contexts. The situative perspective takes the theory of social and ecological interaction as its basis and builds toward a more comprehensive theory by developing increasingly detailed analyses of information structures in the contents of people's interactions". (p. 5) Even if based on other grounds, this viewpoint is compatible with our view that the two positions are not contradictory, but complementary —and not only because SIP focuses on individual design and SIT on collective design!

We judge nevertheless that both Simon and Schön neglect essential aspects of design. Simon disregards the characteristics of the activity discussed in detail in this text that lead to the "rich" nature of design (the specificity of ill-defined problem solving, the role of problem representation building, etc.). Schön gives detailed descriptions of extremely "rich" situations, but neither systematises his observations, nor makes the reflective step that we consider critical and neces-

sary if one wants to attain a "higher" level on which statements about design "in general" can be made.
The switch of focus in design studies, from individual to collaborative design, has also entailed an evolution of the theoretical frameworks. The purely cognitive framework based on the SIP approach is not sufficient for modelling individual design, but in order to address issues related to the collective nature of work, it is certainly inadequate.

5.2 Design as problem solving

Many authors remark that design is no "problem solving". Others consider that design is *not only* problem "solving", but also problem "setting", "formulation", "structuring", "framing", "defining", to name just a few. For a "naive" user, the term "problem solving" may indeed seem inappropriate as applied to design. As we saw however in the Preliminary definitions section, the technical acceptations of "problem" and "problem solving" as used in cognitive psychology, also cover these other problem-centred activities. Modifying or replacing a technical term with a considerable history, i.e. a term that constitutes a key word in psychology, Artificial Intelligence and other disciplines, does not seem reasonable to us.
However, many of these other problem-centred activities indeed do not receive much attention in the SIP problem solving research literature. As noticed already and developed below, we consider that representational activities are the core of design activity —both construction and use of representations, concerning both problem-solution and potentially relevant information.

5.3 Design as ill defined problem solving

We noticed that nearly everywhere in the cognitive design research literature, design is qualified as "ill structured" (or "ill defined") by reference to Simon (1973). As observed in this text, Simon himself did not consider design as specifically "ill structured" —chess and symbolic logic problems were also. If the reference is nevertheless appropriate, it is because of Simon (1973)'s discussion of the more or less structured character of design —and other problem solving activities.
According to Dasgupta (2003, p. 702), "Simon's corpus of work was almost entirely in the realm of ill-structured problems…: administrative decision making, economic decision making, human problem solving, scientific discovery, and design are all examples par excellence of the ill-structured". Dasgupta declares explicitly that "the realm of ill-structured problems" is to be taken "in the sense [Simon (1973)] himself has defined this concept" (2003, p. 702). Adopting this Simon's approach to ill-structured problems makes Dasgupta's position rather different from ours. We indeed consider that human problem solving in these various domains pertains to "ill-defined problem solving", but not in Simon's sense! People who have to solve problems in administrative or economic decision making, designers and scientists making discoveries don't first "structure" their problems, and then "solve" them —as Simon proposes for ill-structured problems.

5.4 Different types of design

According to Thomas and Carroll, "activities as diverse as software design, architectural design, naming and letter-writing appear to have much in common". (Thomas & Carroll, 1979/1984) "Generic" design is a frequently encountered topic —however, never based on comparative cognitive analyses (but see Goel, 1994).
We suppose, however, that the nature of the design object (artefact) influences the design activity (see e.g. our section on The specificity of HCI design). This means that there are different "types" or "forms" of design. As far as we know, this assumption has never been the object of specific cognitive analysis.
Our discussion starts by an interrogation concerning Simon's position. Did Simon think that, depending on the nature of the design object, the cognitive processes and representations involved differ? If he did so, this might explain, at least in part, his different approaches to "design, in general" (exemplified by engineering and architectural design) and social planning.

Would engineering designers proceed in a different way from social planners? Alternatively, is it "simply" a question of size? Or a question of the more or less well or ill definedness of problems? Does Simon perhaps consider that engineering problems are inherently well defined, and social design problems ill defined?
In his *Sciences of the artificial* (1999), Simon proposed a new vision on science, considering these sciences *of the artificial*, these sciences of design, as sciences in their own right, distinct from the natural sciences, which traditionally are considered "the" "Science". Simon's conception of design translates, however, (also) a very conventional view: management and administrative decisions, i.e. actions in the social domain, are fuzzy, imprecise and require a qualitative approach, whereas engineering and "typical" design are precise, exact and can be handled with quantitative methods!
It is mostly in informal discussions that the influence of the type of design object on the activity has been dealt with (Ullman et al., 1988) (Löwgren, 1995). In this section, we will start a reflection on this topic, identifying and discussing some dimensions that may lie beneath differences between types of design.
NOTE. Hubka and Eder (1987), two engineering design methodologists, assert that "decisive for different kinds of design to exist is the *ob□ect* of a design activity, what is being designed, which substantially influences the design process". (p. 124). The authors, however, do not present any evidence, be it empirical or theoretical, for this position. They distinguish four hierarchical levels of design, "and consequently four levels of *ob□ects*". At the most abstract level is "design"; at the second level, design for particular "classes of objects or entities", such as "textile design", "engineering design", "picture, book design"; at the next level, design for "branches" of each one of these classes, e.g., different types of engineering design, such as "mechanical engineering design" (MED), "electrical engineering design", and "civil engineering design"; at the two most specific levels[46], design for "technical system sub-classes", e.g. different types of MED, such as "MED of pumps", "MED of cranes", "MED of locomotives", and, e.g. different types of MED of locomotives, such as "MED of steam locomotives", "MED of Diesel locomotives", "MED of electric locomotives".

Some candidates for dimensions that may differentiate "types" of design, are the following.

- *dependencies between function and form.* A first distinction opposes domains where function and form can be aligned, to domains where individual forms are devised to do many functions simultaneously. In the first type of domain, to each particular form corresponds a particular function (e.g., software design). Mechanical design is an example of the second type. In the first type of domain, a functional decomposition corresponds directly to a form decomposition. In the second type of domain, each design decision can potentially affect every subsequent decision, because a goal may be achieved by modifying a previously specified form, rather than by introducing a new form (Ullman et al., 1988).
- *the nature of the artefact's "material"*. On this dimension, Löwgren opposes what he calls "external" software design (i.e., "design of the external behavior and appearance of the product, the services it offers to users and its place in the organization" in contrast to "internal" design, i.e. "the construction of the software") to other types of design (e.g. architectural and engineering design). Unlike in other design disciplines, in external software design "it is technically possible to evolve a software prototype into a final product". Therefore, "the 'distance' between the design concept and the final product is shorter than in, say, architecture" (p. 93). This does not imply, however, that in the domain of software design, design and implementation are not separated.

For Pahl and Beitz, "software engineering products require systematic design steps similar to those for physical products, but involve only relationships between data elements" (1996).

[46] In their Figure 2 (p. 125), the authors seem to present five, rather than four, levels.

- *the temporal nature of the artefact*. "Interactive systems are designed to have a certain behavior over time, whereas houses typically are not", according to Löwgren (1995, p. 94). Even if this assertion is questionable, the "temporal nature" of artefacts, especially their evolutive character, is a dimension on which artefacts may differ. Even if houses do not have"behaviour" over time, they generally evolve over time. "Good" architects do anticipate the evolution that their design may undergo, e.g. through its "use" (see also Simon's approach to social planning). This characteristic of artefacts is to be distinguished from what Carroll, Rosson, Chin and Koenemann (1997, p. 63) call the "transformative" nature of certain systems. Carroll and al. observe that the kind of systems they are developing (enabling novel educational activities) are "transformative", in the sense of "fundamentally [altering] possibilities for human behavior and experience".
- *the type of evaluation that is possible*, and consequently the possibilities for comparing different design proposals. Domains differ in the types of measures, and other means, that can be used in order to evaluate design solutions (Malhotra et al., 1980, pp. 129-130). In engineering, e.g., the civil design of a bridge, "objective" measures of future artefacts' performance can be used. One can calculate, e.g., whether the proposed designs meet functional requirements such as accommodation and maximum load. Using such measures, different solution proposals can be ranked in some manner, even if subjective criteria, such as aesthetics, which are more difficult to integrate in one combined "score", may also play a role. In software design, performance measures can be defined, but as long as the design is not implemented, it cannot be tested. In the domain of software design, "it is very difficult, in fact, to ascertain whether the design is complete, is consistent, or whether it meets the functional requirements. This seems equivalent to not being able to tell if a building will stand up!" (Malhotra et al., 1980, p. 129)
- *delay of implementation*. Something that has been considered to make quite difficult the solving of social-science problems, is the "delay from the time a solution is proposed and accepted to when it is fully implemented" (Voss et al., 1983). This factor influences in particular evaluation of solutions. "Naturally, a good solution anticipates changes in conditions, but anticipation can be quite difficult". (ibid.) This remark that was made with respect to social-science problems, undoubtedly also holds for other design problems.
- *"designing in space versus time"* (Thomas & Carroll, 1979/1984). We saw above, in the section "Temporal and spatial constraints", that these constraints are dealt with differently, but the difficulty of processing spatial compared to temporal attributes and its underlying factors, have not yet been settled unambiguously.

This list of candidates for dimensions that might differentiate "types" of design is a start for discussion and analysis of such dimensions. A further step would be to elucidate *how* these —and other— differences influence dimensions of design *activity* and/or its *result*, the artefact.

□□□

We have the impression that, among the ever more publications dedicated to Simon since his death in 2001 (see e.g. Dasgupta, 2003; Perrin, 2002; Pitrat, 2002), this text has provided an original contribution, not only to what Dasgupta (2003, p. 703) proposes to call "Simon Studies", but also to design studies and cognitive psychology, and their common field, i.e. cognitive design studies.

As we announced, our own proposals concerning a cognitive design model privilege the blackboard approach for modelling. The integrated model remains to be constructed, guided by the —indirect— constraints and specifications formulated above and those that are shortly going to be developed below, in the final section of this text.

5.5 Our approach to design: conclusion

We defined design as an activity that consists in specifying an artefact, given requirements that indicate one or more functions to be fulfilled and/or objectives to be satisfied by that artefact.

The activity of design thus consists in *transforming representations*. Design indeed starts with a representation and has to come up with another representation (both, generally, composite). The initial representation can be very diverse, i.e. composed of elements of various levels, from different sources, made up of contradictory and/or incomplete constraints, or implying such elements. The final representation has to be very precise and detailed, i.e. composed of elements that are all at the same level of abstraction: it has to be so specific that the implementation of the artefact is completely specified.

Most features presented in this text as characteristic of design problems contribute to complicate this representational activity: their ill definedness, complexity and ambiguity, and the incomplete, and especially the conflicting nature of their constraints.

In an activity that has representations as its object, knowledge and other representations play of course a central role. Even if designers obviously need general, abstract knowledge and weak, generally applicable methods, domain-specific knowledge, and the corresponding strong methods are essential. We suppose that satisficing, e.g., requires more domain-specific knowledge than optimising does. Likewise for what we called, with Archer (1965/1984), the "creative" aspect of design —which we do not at all oppose to "problem solving". Knowledge is also a key element in the exercise of analogical reasoning —which may, besides, be a factor of creativity (see Johnson-Laird, 1989's "nondeterministic leaps").

Recognition plays an important role in all these activities —and Simon was completely right in highlighting its role, besides that of search. In our opinion, Simon overestimated its importance compared to the conscious use of knowledge —the one that Schön *et al.* emphasised. In order to recognise, and also in oder to evoke through analogical reasoning, a potentially relevant element of knowledge, memory associations must exist between target (feature of the situation) and source (knowledge element). Otherwise, conscious search or exploration has to be used, sometimes leading to "surprise" (Simon) and other "discovery of unintended consequences" (Schön).

In the case of a professional activity as design, the knowledge on which recognition, analogical reasoning and other knowledge-intensive activities are based, is grounded mainly in professional experience (but not only, see Visser, 1995b). Knowledge also shapes the ill definedness of a problem (but in the inverse sense, we suppose, as advanced by Simon, 1973/1984, p. 197 quoted above).

All activities based on knowledge have an evolving nature with respect to their "problem" character, their "routine" character, their "ill-definedness" —and possibly even their "design" character. This is indeed a likely consequence of our defining a person's task and activity as "design" based on cognitively based characteristics, themselves determined by the person's knowledge.

Adopting an entirely "problem-solv*er*-oriented" approach, Thomas and Carroll (1979/1984, p. 222) consider design as a "way of looking at" a problem and therefore not restricted to a particular "type" of problem. "Any problem can be looked at as a design problem". (ibid.) Theorem proving, e.g., can be *viewed* as designing if, say, the requirement to stay inside certain formal rules is relaxed and creativity is allowed; in that case, although the initial goal was to prove a theorem, the goal actually becomes "to find out something interesting". In the same way, "designing" a house by applying a set of standard rules to the stated requirements is no longer "design" in a cognitive-activity oriented acceptation. Therefore, whether a task is "design" depends entirely on the problem solver. "Much of what [people] call technological progress may be viewed as a process of rendering ill-structured design problems as more well-structured procedures for accomplishing the same ends —without requiring design". (Thomas & Carroll, 1979/1984, p. 222)

Indeed, if a "design" task is no longer "ambiguous" (see Reitman, 1964), if its constraints are the object of an agreement —possibly implicit, be it social, societal or based on enterprise policy—, a "design problem" can become a "transformation problem" —or even constitute no longer a "problem" at all!

6 References

Adelson, B. (1984). When Novices Surpass Experts : The Difficulty of a Task may Increase With Expertise. *□ournal of Experimental Psychology □□earning, □emory and Cognition, 1*□(3), 483-495.

Adelson, B., Littman, D., Ehrlich, K., Black, J., & Soloway, E. (1985). Novice-expert differences in software design. In B. Shackel (Ed.), *Human-computer interaction - I□□E□-AC□'8*□. Amsterdam: North-Holland.

Adelson, B., & Soloway, E. (1988). A model of software design. In M. T. H. Chi, R. Glaser & M. J. Farr (Eds.), *□he nature of expertise* (pp. 185-208). Hillsdale, NJ: Lawrence Erlbaum Associates.

Agre, P. (1993). The symbolic worldview: Response to Vera and Simon. *Cognitive Science, 1*□(1), 61-70.

Agre, P., & Chapman, D. (1987). Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth □ational Conference on Artificial Intelligence* (pp. 268-272). Menlo Park, CA: American Association for Artificial Intelligence.

Akin, O. (1979/1984). An exploration of the design process. In N. Cross (Ed.), *Developments in design methodology* (pp. 189-207). Chichester: Wiley (Originally published in *Design □ethods and □heories*, 1979, *13* (3/4), 115-119.).

Akin, □. (1986). *Psychology of architectural design*. London: Pion.

Akin, □. (1992). A structure and function based theory for design reasoning. In N. Cross, K. Dorst & N. Roozenburg (Eds.), *□esearch in design thinking*. Delft: Delft University Press.

Alexander, C. (1963/1984). The determination of components for an Indian village. In N. Cross (Ed.), *Developments in design methodology* (pp. 33-56). Chichester: Wiley (Originally published in J. C. Jones and D. Thornley (Eds.) (1963). Conference on design methods. Oxford: Pergamon.).

Anderson, J. R. (1983). *□he architecture of cognition*. Cambridge, MA: Harvard University Press.

Anderson, J. R. (Ed.). (1976). *□anguage, memory, and thought*. Hillsdale, NJ: Erlbaum.

Anderson, J. R., Reder, L. M., & Simon, H. A. (1996). Situated learning and education. *Educational □esearcher, 2*□(4), 5-11.

Anderson, J. R., Reder, L. M., & Simon, H. A. (January/February 1997). Rejoinder: Situated versus cognitive perspectives: Form versus substance. *Educational □esearcher, 26*(1), 18-21.

Anzai, Y., & Simon, H. A. (1979). The theory of learning by doing. *Psychological □eview, 86*, 124-140.

Archer, L. B. (1965/1984). Systematic method for designers. In N. Cross (Ed.), *Developments in design methodology* (pp. 57-82). Chichester: Wiley (Originally published by The Design Council, London, 1965).

Asimov, M. (1962). *Introduction to design*. Prentice-Hall: Englewood Cliffs, NJ.

Ball, L. J., Evans, J. S. B. T., & Dennis, I. (1994). Cognitive processes in engineering design: A longitudinal study. *Ergonomics, 3*□(11), 1753-1786.

Ball, L. J., & Ormerod, T. C. (1995). Structured and opportunistic processes in design: A critical discussion. *International □ournal of Human-Computer Studies, □3*, 131-151.

Ball, L. J., & Ormerod, T. C. (2000). Putting ethnography to work: The case for a cognitive ethnography of design. *International □ournal of Human-Computer Studies, □3*, 147-168.

Balzer, R. (1981). Transformation implementation: an example. *IEEE □ransactions on Software Engineering, SE-□*, 3-14.

Bardram, J. (1997, September). *Plans as Situated Action□An Activity □heory approach to workflow systems.* Paper presented at the Proceedings of ECSCW'97 Conference, Lancaster UK.

Barstow, D. R. (1984). A perspective on automatic programming. □*he AI □aga□ine,* □.

Basili, V. R., & Turner, A. J. (1975). Iterative enhancement: a practical technique for software development. *IEEE □rans. Software Engineering, 1*(4), 390-396.

Baykan, C. (1996). Design strategies.

Béguin, P. (1994). □*ravailer avec la C.A.O. en ing□nierie industrielle □de l'individuel au collectif dans les cativit□s avec instruments.* Unpublished Thèse de Doctorat, spécialité Ergonomie, CNAM, Paris.

Benyon, D. (1992). Task Analysis and System Design: The Discipline of Data. *Interacting with Computers,* □(2), 246 - 259.

Best, B. J., & Simon, H. A. (2000). Simulating human performance on the traveling salesman problem. In N. Taatgen & J. Aasman (Eds.), □*hird International Conference on Cognitive □odeling* (pp. 42-49). Groningen, Netherlands: Universal Press.

Bhaskar, R., & Simon, H. A. (1977). Problem solving in semantically rich domains: An example from engineering thermodynamics. *Cognitive Science, 1*, 193-215.

Biggerstaff, T. J., & Perlis, A. J. (Eds.). (1989a). *Software reusability* (Vol. 2). Reading, MA: Addison-Wesley.

Biggerstaff, T. J., & Perlis, A. J. (Eds.). (1989b). *Software reusability□Concepts and □odels* (Vol. 1). New York, NY: ACM press.

Bisseret, A. (1990). Towards computer-aided text production. In P. Falzon (Ed.), *Cognitive ergonomics□□nderstanding, learning and designing human-computer interaction*. Londres: Academic press.

Bisseret, A., Figeac-Letang, C., & Falzon, P. (1988). □*odeling opportunistic reasonings□the cognitive activity of traffic signal setting technicians* (Rapport de recherche No. 898). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Blessing, L., Brassac, C., Darses, F., & Visser, W. (Eds.). (2000). *Analysing and modelling collective design activities. Proceedings of COOP 2□□, □ourth International Conference on the Design of Cooperative Systems*. Rocquencourt: Institut National de Recherche en Informatique et en Automatique.

Blessing, L. T. M. (1994). *A process-based approach to computer-supported engineering design.* Universiteit Twente, Enschede.

Bloch, H., Chemama, R., Gallo, A., Leconte, P., Ny, J.-F. L., Postel, J., et al. (Eds.). (1991). □*rand dictionnaire de la psychologie*. Paris: Larousse.

Boehm, B. (1976). Software Engineering. *IEEE □rans. Computer, C-2*□(12), 1226-1241.

Boehm, B. W. (1981). *Software engineering economics*. Englewood Cliffs: Prentice Hall.

Boehm, B. W. (1986). A spiral model of software development and enhancement. *AC□ SI□-SO□□Software Engineering □otes, 11*(4), 22-32.

Bonnardel, N. (1989). □'□*valuation de solutions dans la resolution de probl□mes de conception* (Rapport de recherche No. 1072): Institut National de Recherche en Informatique et en Automatique.

Bonnardel, N. (1991a). *Criteria used for evaluation of design solutions.* Paper presented at the Designing for Everyone and Everybody: 11th Congress of the International Ergonomics Association, London.

Bonnardel, N. (1991b). L'évaluation de solutions dans la résolution de problèmes de conception et dans les systèmes experts critiques. In D. Hérin-Aime, R. Dieng, J. P. Regouard & J. P. Angoujiard (Eds.), □*nowledge □odeling □ Expertise □ransfer*. Amsterdam, Washington, Tokyo: I.O.S. Press.

Bonnardel, N. (1992). □*e r□le de l'□valuation dans les activit□s de conception.* Unpublished Thèse de doctorat, Université de Provence.

Breuker, J., Wielinga, B., van Someren, M., de Hoog, R., Schreiber, G., de Greef, P., et al. (1987). *odel-driven knowledge acquisition interpretation models* (No. Deliverable task A1, Esprit Project 1098). Amsterdam: University of Amsterdam.

Brooks, R. (1977). Towards a theory of the cognitive processes in computer programming. *International ournal of an- achine studies, 9*, 737-751.

Brown, D., & Chandrasekaran, B. (1989). *Design problem solving. nowledge structures and control strategies*. London: Pitman.

Brown, J. S., Collins, A., & Duguid, P. (1989). Situated cognition and the culture of learning. *Educational esearcher, 18*(1, Jan-Feb), 32-42.

Bruner, J. S., Goodnow, J. J., & Austin, G. A. (1956). *A study of thinking*. New York: Wiley.

Bucciarelli, L. (1988). An ethnographic perspective on engineering design. *Design Studies, 9*(3), 159-168.

Bucciarelli, L., & Sharville, S. (1996). Designing engineers. *Design Studies, 1* (2), 221.

Bucciarelli, L. L. (1984). Reflective practice in engineering design. *Design Studies,* (3), 185-190.

Bucciarelli, L. L. (2002). Between thought and object in engineering design. *Design Studies, 23*(5), 219–231.

Buchanan, R. (1990, 17-19 octobre 1990). *he "icked Problems" theory of design.* Paper presented at the Colloque Recherches sur le Design, Compiègne.

Buchanan, R. (Spring 1992). Wicked Problems in Design Thinking. *Design Issues, 8*(2), 5-.

Buchanan, R., & Margolin, V. (Eds.). (1995). *Discovering Design Explorations in Design Studies*. Chicago: University of Chicago Press.

Burkhardt, J.-M. (1997). *utilisation de solutions en conception orient e-ob et un mod le cognitif des m canismes et repr sentations mentales.* Unpublished Thèse de Doctorat en Psychologie Cognitive / Ergonomie, Université René Descartes de Paris V, Paris.

Burkhardt, J.-M., & Détienne, F. (1995). La réutilisation de solutions en conception de programmes informatiques. *Psychologie ran aise, Sp cial "'Ergonomie Cognitive",* (1), 85-98.

Burns, C., & Vicente, K. J. (1995). A framework for describing and understanding interdisciplinary interactions in design. In *DIS 9*

Byrne, R. (1977). Planning meals: Problem-solving on a real data-base. *Cognition,* , 287-332.

Cagan, J., Kotovsky, K., & Simon, H. A. (2001). Scientific discovery and inventive engineering design: Cognitive and computational similarities. In E. K. Antonsson & J. Cagan (Eds.), *ormal engineering design synthesis* (pp. 442-465). New York, NY: Cambridge University Press.

Campbell, M., Cagan, J., & Kotovsky, K. (1999). A-Design: an agent-based approach to conceptual design in a dynamic environment. *esearch in Engineering Design, 11*, 172-192.

Campbell, M., Cagan, J., & Kotovsky, K. (2000). Agent-based synthesis of electro-mechanical design configurations. *AS E ournal of echanical Design, 122*(1), 61-69.

Card, S., Moran, T. P., & Newel, A. (Eds.). (1983). *he psychology of human-computer interaction*. Hillsdale, NJ: Erlbaum.

Carroll, J. M. (1995). *Scenario-based design. Envisioning work and technology in system development*. Hillsdale, NJ: Erlbaum.

Carroll, J. M. (2000). *aking use. Scenario-based design of human computer interactions*. Cambridge, MA: The MIT Press.

Carroll, J. M., & Rosson, M. B. (1985). Usability specifications as a tool in iterative development. In H. R. Hartson (Ed.), *Advances in human-computer interaction* (Vol. 1). Norwood, NJ (USA): Ablex.

Carroll, J. M., Rosson, M. B., Chin, G., & Koenemann, J. (1997). Requirements development: stages of opportunity for collaborative needs discovery. In *DIS'9*

Carroll, J. M., Thomas, J. C., & Malhotra, A. (1980). Presentation and representation in design problem-solving. *British Journal of Psychology, 71*, 143-153.

Carroll, J. M., Thomas, J. C., Miller, L. A., & Friedman, H. P. (1980). Aspects of structure in design problem solving. *American Journal of Psychology, 93*, 269-284.

Chalmé, S., Visser, W., & Denis, M. (2000, 6-7 Septembre). *Cognitive aspects of urban route planning.* Paper presented at the Proceedings of ICTTP 2000 (International Conference in Trafic and Transport Psychology) [CD-ROM], Berne (Suisse).

Chalmé, S., Visser, W., & Denis, M. (in press). Cognitive effects of environmental knowledge on urban route planning strategies. In T. Rothengatter & D. Huguenin (Eds.), *Traffic and Transport Psychology*. Amsterdam: Elsevier.

Chatel, S., & Détienne, F. (1996). Strategies in object-oriented design. *Acta Psychologica, 91*, 245-269.

Clancey, W. (1985). Heuristic classification. *Artificial Intelligence, 27*, 289-350.

Clancey, W. J. (1991). Situated cognition: Stepping out of representational flatland. *AI Communications - The European Journal on Artificial Intelligence, 4*(2/3), 109-112.

Clancey, W. J. (1993). Situated action: A neuropsychological interpretation. Response to Vera and Simon. *Cognitive Science, 17*(1), 87-116.

Cook, C. R., Scholtz, J. C., & Spohrer, J. C. (Eds.). (1993). *Empirical Studies of Programmers - Fifth Workshop*. New Brunswick, NJ (USA): Ablex.

Cross, N. (1984). *Developments in design methodology*: Wiley.

Cross, N. (1984). Introduction to Part One: The Management of design process. In N. Cross (Ed.), *Developments in design methodology* (pp. 1-7). Chichester: Wiley.

Cross, N., Christiaans, H., & Dorst, K. (1997). Analysing design activity. *Design Studies, 18*(4), 75-322.

Cross, N., Christiaans, H., & Dorst, K. (Eds.). (1996). *Analysing design activity*. Chichester: Wiley.

Cross, N., Dorst, K., & Roozenburg, N. (Eds.). (1992). *Research in design thinking*. Delft: Delft University Press.

Culverhouse, P. F. (1995). Constraining designers and their CAD tools. *Design Studies, 16*(1), 81-101.

Curtis, B. (1986). *By the way, did anyone study any real programmers* Paper presented at the Empirical Studies of Programmers: First workshop, Norwood, NJ (USA).

D'Astous, P., Détienne, F., Visser, W., & Robillard, P. N. (in press). Changing our view on design evaluation meetings methodology: a study of software technical review meetings. *Design Studies*.

D'Astous, P., Robillard, P., Détienne, F., & Visser, W. (2001). Quantitative measurements of the influence of participant roles during peer review meetings. *Empirical Software Engineering, 6*, 143-159.

Daniellou, F. (1999). *Le statut de la pratique et des connaissances dans l'intervention ergonomique de conception (Texte de l'Habilitation à Diriger des Recherches présenté en 1992, complété d'un Avant-propos, un complément de bibliographie et de deux articles sur le même thème).* Université Victor Segalen Bordeaux 2 - ISPED, Bordeaux.

Darke, J. (1979/1984). The primary generator and the design process. In N. Cross (Ed.), *Developments in design methodology* (pp. 175-188). Chichester: Wiley (Originally published in *Design Studies*, 1979, *1* (1), 36-44.).

Darses, F. (1990a, September, 2-6). *An assessment of the constraint satisfaction approach for design: A psychological investigation.* Paper presented at the ECCE5, Urbino, Italie.

Darses, F. (1990b, November 20-23). *Constraints in design: An empirical study.* Paper presented at the Cognitiva 90, Madrid, Spain.

Darses, F. (1990c). *Constraints in design: towards a methodology of psychological analysis based on AI formalisms.* Paper presented at the INTERACT'90, North Holland.

Darses, F. (1990). *Gestion de contraintes au cours de la resolution d'un problème de conception de reseaux informatiques* (Rapport de recherche No. 1164). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Darses, F. (1994). *Gestion des contraintes dans la résolution des problèmes de conception.* Unpublished Thèse de Doctorat, spécialité Psychologie Cognitive, Université Paris 8, Saint-Denis.

Darses, F. (1997). L'ingénierie concourante: Un modèle en meilleure adéquation avec les processus cognitifs en conception. In P. Brossard, C. Chanchevrier & P. Leclair (Eds.), *Ingénierie Concourante. De la technique au social*. Paris: Economica.

Darses, F. (2002, 23-25 June 2002). *A cognitive analysis of collective decision-making in the participatory design process.* Paper presented at the Proceedings of the Seventh Participatory Design Conference, Malmö (Sweden).

Darses, F., Détienne, F., Falzon, P., & Visser, W. (September 2001). *CODE: A method for analysing collective design processes* (Research report INRIA No. n° 4258). Rocquencourt: Institut National de Recherche en Informatique et en Automatique.

Dasgupta, S. (1989). The structure of design processes. *Advances in Computers, 28*, 1-67.

Dasgupta, S. (2003). Multidisciplinary Creativity: The Case of Herbert A. Simon. *Cognitive Science, 2*(5), 683-708.

De Vries, E. (1994). *Structuring information for design problem solving (Ph.D. thesis)*. Den Haag: Koninklijke Bibliotheek.

Depraz, N., Varela, F., & Vermersch, P. (2003). *On becoming aware: steps to a phenomenological pragmatics*. Amsterdam: John Benjamins.

Design Studies. (1997). Special issue on Descriptive models of design. *Design Studies, 18*(4).

Desnoyers, L., & Daniellou, F. (1989). *SELF: the Francophone Ergonomics Society*. Retrieved 23/01/04, from www.ergonomie-self.org/Pages/self/presentation/desnoyers.html

Détienne, F. (1991a). *Reusing solutions in software design activity: An empirical study.* Paper presented at the CHI'91, New Orleans.

Détienne, F. (1991b). *Solution reuse in expert design activity.* Paper presented at the International conference on cognitive expertise, University of Aberdeen, UK.

Détienne, F. (1994). Constraints on design: language, environment, code representation. In Gilmore, R. Winder & Détienne (Eds.), *User centred requirements for software engineering environments*: Springer Verlag, NATO ASI Series.

Détienne, F. (1998). *Génie logiciel et psychologie de la programmation*. Paris: Hermès.

Détienne, F. (2002a). *Software design. Cognitive aspects*. London: Springer.

Détienne, F. (2002b, June 18-21). *Supporting collaborative design: Current research issues.* Paper presented at the Fourteenth annual workshop of the Psychology of Programming Interest Group (PPIG14), London (U.K.).

Détienne, F., & Burkhardt, J.-M. (2001). Des aspects d'ergonomie cognitive dans la réutilisation en génie logiciel. *Techniques et Sciences Informatiques, 2*(4), 461-487.

Détienne, F., Burkhardt, J.-M., & Visser, W. (2003). Cognitive effort in collective software design: methodological perspectives in cognitive ergonomics. In A. Jedlitschka & M. Ciolkowski (Eds.), *Proceedings of the ESEIW 2003 Workshop on Empirical Studies in Software Engineering (ESESE 2003), 2nd Workshop in the Workshop Series on Empirical Software Engineering "The Future of Empirical Studies in Software Engineering"* (pp. 17-25). Roman Castles (Italy).

Détienne, F., & Falzon, P. (2001). Cognition and Cooperation in Design: the Eiffel research group. In M. Hirose (Ed.), *Human-Computer Interaction - Interact 2001* (pp. 879-880): IOS Press.

Dillon, A., & Sweeney, M. (1988, 5-9 sept.). *The application of cognitive psychology to CAD.* Paper presented at the People and Computers IV, Manchester.

Donmoyer, R. (1996, May). Introduction. This issue: a focus on learning. *Educational Researcher, 2*(4), 4.

Dörner, D. (1999). Approaching design thinking research. *Design Studies, 2*(5), 407-416.
Dorst, K. (1997). *Describing design. A comparison of paradigms.* Technische Universiteit Delft, Delft.
Dwarakanath, S., & Blessing, L. (1996). Ingredients of the design process: a comparison between group and individual work. In N. Cross, H. Christiaans & K. Dorst (Eds.), *Analysing design activity* (pp. 93-116). Chichester: Wiley.
Dykstra-Erickson, E., Mackay, W., & Arnowitz, J. (2001). design *(of)*. *interactions, march-april*, 109-117.
Eastman, C. (1969). *Cognitive processes and ill-defined problems a case study of design.* Paper presented at the IJCAI'69, International Joint Conference on Artificial Intelligence, Washington, D.C.
Eastman, C. (1970). On the analysis of intuitive design processes. In G. T. Moore (Ed.), *Emerging ethods in Environmental Design and Planning. Proceedings of the irst International Design ethods Conference* (pp. 21-37). Cambridge, MA: The MIT Press.
Empirical Studies of Programmers (ESP) Seventh orkshop (Ed.). (Eds.). (1997).
Empirical Studies of Programmers Eighth orkshop. (1999).
Ericsson, K. A., & Simon, H. A. (1980). Verbal reports as data. *Psychological eview, 8*, 215-251.
Ericsson, K. A., & Simon, H. A. (1984). *Protocol Analysis. erbal eports as Data* (revised 1993 ed.). Cambridge, MA: The MIT Press.
Falzon, P., & Visser, W. (1989). Variations in expertise: implications for the design of assistance systems. In G. Salvendy & M. J. Smith (Eds.), *Designing and using human-computer interfaces and knowledge based systems* (Vol. II). Amsterdam: Elsevier.
Feitelson, J., & Stefik, M. J. (1977). *A case study of the reasoning in a genetics experiment* (No. Rep. N° HPP-77-18). Stanford: Stanford University, Computer Science Department, Heuristic Programming Project.
Ford, S., & Marchak, F. M. (January 1997). The Future of Visual Interaction Design? *SI CHI Bulletin, 29*(1).
Frankenberger, E., & Badke-Schaub, P. (1999). Special Issue: Empirical Studies of Engineering Design in Germany - Editorial. *Design Studies, Special Issue Empirical Studies of Engineering Design in ermany, 2*(5), 397-400.
French, M. J. (1971). *Engineering design he conceptual stage*. London: Heinemann.
Fricke, G. (1999). Successful approaches in dealing with differently precise design problems. *Design Studies, Special Issue Empirical Studies of Engineering Design in ermany, 2*(5), 417-430.
Galle, P. (1996). Replication protocol analysis: a method for the study of real world design thinking. *Design Studies, 1*(2), 181-200.
Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995a). *Design patterns. Elements of reusable ob ect-oriented software*. Reading, MA: Addison-Wesley.
Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995b). Element of Reusable object-oriented Software. In *Design Patterns*. massachusetts: Addison-Wesley Publishing Company.
Gero, J. (1998). Towards a model of designing which includes its situatedness. In H. Grabowski, S. Rude & G. Grein (Eds.), *niversal design theory* (pp. 47-56). Aachen (Germany): Shaker Verlag.
Gero, J. (Ed.). (1991). *Artificial Intelligence in Design '91*. Oxford: Butterworth-Heinemann.
Gero, J. S. (1990). Design prototypes: a knowledge representation schema for design. *AI aga ine, 11*(4), 26-36.
Gero, J. S. (1998). Conceptual designing as a sequence of situated acts. In I. Smith (Ed.), *Artificial Intelligence in structured engineering* (pp. 165-177). Berlin: Springer.
Gero, J. S. (Ed.). (1992). *Artificial Intelligence in Design '92*. Boston: Kluwer.

Gero, J. S., & McNeill, T. (1998). An approach to the analysis of design protocols. *Design Studies, 19*(1), 21-61.

Gick, M. L., & Holyoak, K. (1983). Schema Induction and Analogical Transfer. *Cognitive Psychology, 1*□, 1-38.

Gick, M. L., & Holyoak, K. J. (1980). Analogical problem solving. *Cognitive Psychology, 12*, 306-355.

Gilhooly, K. J. (1989). Human and machine problem solving: toward a comparative cognitive science. In K. J. Gilhooly (Ed.), *Human and machine problem solving* (pp. 1-12). New York: Plenum.

Gilmore, D. (1990). Methodological issues in the study of programming. In J. M. Hoc, T. Green, R. Samur□ay & Gilmore (Eds.), *Psychology of programming*. London: Academic Press.

Gilmore, D., Winder, R., & Détienne, F. (Eds.). (1994). □*ser-centred requirements for software engineering environments*. Berlin: Springer.

Goel, V. (1994). A comparison of design and nondesign problem spaces. *Artificial Intelligence in Engineering, 9*, 53-72.

Goel, V., & Pirolli, P. (1992). The structure of design problem spaces. *Cognitive Science, 16*, 395-429.

Gray, W. D., & Boehm-Davis, D. A. (Eds.). (1996). *Empirical Studies of Programmers□Sixth □orkshop*. New Brunswick, NJ (USA): Ablex.

Green, T. R. G. (1980). Programming as a cognitive activity. In H. T. Smith & T. R. G. Green (Eds.), *Human interaction with computers*. London: Academic Press.

Green, T. R. G. (1990). Programming languages as information structures. In Hoc, T. R. G. Green, R. Samur□ay & Gilmore (Eds.), *Psychology of Programming* (pp. 117-137): Academic Press.

Greeno, J. G. (1978). Natures of problem-solving abilities. In W. K. Estes (Ed.), *Handbook of learning and cognitive processes* (Vol. V. Human information processing, pp. 239-270). Hillsdale, NJ: Lawrence Erlbaum.

Greeno, J. G. (1998). The situativity of knowing, learning, and research. *American Psychologist,* □*3*(1), 5-26.

Greeno, J. G. (January/February 1997). Response: On Claims that Answer the Wrong Questions. *Educational □esearcher, 26*(1), 5-17.

Greeno, J. G., & Moore, J. L. (1993). Situativity and symbols: Response to Vera and Simon. *Cognitive Science, 1*□(1), 49-60.

Greeno, J. G., & Simon, H. A. (1988). Problem solving and reasoning. In R. C. Atkinson, R. J. Herrnstein, G. Lindzey & R. D. Luce (Eds.), *Stevens' handbook of experimental psychology* (2 ed., Vol. II, pp. 589-672). New York, NY: Wiley.

Guindon, R. (1990). Designing the design process: exploiting opportunistic thoughts. *Human Computer Interaction,* □, 305-344.

Guindon, R. (1990). Knowledge exploited by experts during software System Design. *International □ournal of □an-□achine Studies, 33*, 279-304.

Guindon, R., Krasner, H., & Curtis, B. (1987). Breakdowns and processes during the early activities of software design by professionals. In G. Olson, S. Sheppard & E. Soloway (Eds.), *Empirical Studies of Programmers□Second □orkshop*. Norwood, NJ (USA): Ablex.

Günther, J., Frankenberger, E., & Auer, P. (1996). Investigation of individual and team design processes. In N. Cross, H. Christiaans & K. Dorst (Eds.), *Analysing design activity* (pp. 117-132). Chichester: Wiley.

Hamel, R. (1989). Design process and design problems in architecture. □*ournal of Environmental Psychology, 9*, 73-77.

Hayes, J. R., & Flower, L. S. (1980). Identifying the organization of writing processes. In L. W. Gregg & E. R. Steinberg (Eds.), *Cognitive processes in writing*. Hillsdale, NJ: Erlbaum.

Hayes-Roth, B., & Hayes-Roth, F. (1979). A cognitive model of planning. *Cognitive Science, 3*, 275-310.

Hayes-Roth, B., Hayes-Roth, F., Rosenschein, S., & Cammarata, S. (1979). *Modeling planning as an incremental, opportunistic process.* Paper presented at the 6th International Joint Conference on Artificial Intelligence, Tokyo.

Hesse, M. (1988). Theories, family resemblances and analogy. In D. H. Helman (Ed.), *Analogical reasoning Perspectives of artificial intelligence, cognitive science, and philosophy*. Boston: Reidel.

Hoc, J.-M., Green, T. R. G., Samurçay, R., & Gilmore, D. (Eds.). (1990). *Psychology of programming*. London: Academic Press.

Hoc, J. M. (1987). *Psychologie cognitive de la planification*. Grenoble: Presses Universitaires de Grenoble.

Hoc, J. M. (1988). *Cognitive psychology of planning*. London: Academic Press.

Hollan, J., Hutchins, E., & Kirsh, D. (June 2000). Distributed cognition: Toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human interaction,* (2), reprinted in Carroll, John (2000). Human-computer interaction in the new millennium (pp. 2075-2094).

Holyoak, K. J. (1984). Analogical Thinking and Human Intelligence. In R. J. Sternberg (Ed.), *Advances in the Psychology of Human Intelligence* (Vol. 5, pp. 199-230).

Hsu, W., & Liu, B. (2000). Conceptual design: issues and challenges. *Computer-Aided Design, 32*(14), 849-850.

Hubka, V., & Eder, W. E. (1987). A scientific approach to engineering design. *Design Studies, 8*(3), 123-137.

Hutchins, E. (1995). How a cockpit remembers its speed. *Cognitive Science, 19*(3), 265-288.

Hutchins, E. (1996). *Cognition in the wild*. Cambridge, MA: The MIT Press.

Jackson, M. A. (1975). *Principles of program design*. New York: Academic Press.

Jackson, M. A. (1983). *System development*. Englewood Cliffs: Prentice Hall.

Jedlitschka, A., & Ciolkowski, M. (2003a). *The Future of Empirical Studies in Software Engineering.* Paper presented at the ESEIW 2003 Workshop on Empirical Studies in Software Engineering WSESE 2003, 2nd Workshop in the Workshop Series on Empirical Software Engineering, Roman Castles (Italy).

Jedlitschka, A., & Ciolkowski, M. (Eds.). (2003b). *Proceedings of the ESEIW 2003 Workshop on Empirical Studies in Software Engineering WSESE 2003, 2nd Workshop in the Workshop Series on Empirical Software Engineering "The Future of Empirical Studies in Software Engineering"*. Roman Castles (Italy).

Jeffries, R., Turner, A. A., Polson, P. G., & Atwood, M. E. (1981). The processes involved in designing software. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition* (pp. 255-283). Hillsdale, NJ: Erlbaum.

Johnson-Laird, P. N. (1989). Analogy and the exercise of creativity. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning*. Cambridge: Cambridge University Press.

Jonas, W. (1993). Design as problem-solving? or: here is the solution - what was the problem? *Design Studies, 1*(2), 157-170.

Jones, J. C. (1984, Originally published in 1963). A method of systematic design. In N. Cross (Ed.), *Developments in design methodology*. Chichester: Wiley.

Kant, E. (1985). Understanding and automating algorithm design. *IEEE Transactions on Software Engineering, SE-11*, 1361-1374.

Kant, E., & Newell, A. (1984). Problem solving techniques for the design of algorithms. *Information Processing and Management, 2*(1-2), 97-118.

Kaplan, C. A., & Simon, H. A. (1990). In search of insight. *Cognitive Psychology, 22*, 374-419.
Kapor, M. (1996). A software design manifesto. In Winograd (Ed.), *Bringing design to software* (pp. 1-9). New York: ACM Press.
Karsenty, L. (1991). *Le dialogue de validation d'un schéma conceptuel des données* (Rapport de recherche No. 1551). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.
Keane, M. (1994). Analogical asides on case-based reasoning. In S. Wess & al. (Eds.), *Topics in Case-Based Reasoning. First European Workshop, EWCB-93*: Springer.
Kelley, D., & Hartfield, B. (1996). The designer's stance. In Winograd (Ed.), *Bringing design to software* (pp. 151-170). New York: ACM Press.
Kerbrat-Orecchioni, C. (1990). *Les interactions verbales* (Vol. I. Approches interactionnelle et structure des conversations). Paris: Armand Colin.
Kerbrat-Orecchioni, C. (1992). *Les interactions verbales* (Vol. II). Paris: Armand Colin.
Kerbrat-Orecchioni, C. (1994). *Les interaction verbales* (Vol. III. Variations culturelles et échanges rituels). Paris: Armand Colin.
Kim, J., Javier-Lerch, F., & Simon, H. A. (1995). Internal representation and rule development in object-oriented design. *ACM Transactions on Computer-Human Interaction, 2*(4, December), 357-390.
Kitchenham, B., & Carn, R. (1990). Research and practice: Software design methods and tools. In Hoc, T. R. G. Green, R. Samurçay & Gilmore (Eds.), *Psychology of Programming* (pp. 271-284). London: Academic Press.
Klahr, D., & Simon, H. A. (2001). What have psychologists (and others) discovered about the process of scientific discovery? *Current Directions in Psychological Science, June, 1*(3), 75-79.
Klein, M., & Lu, S. (1989). Conflict resolution in cooperative design. *Artificial Intelligence in Engineering, 4*, 168-180.
Koenemann-Belliveau, J., Moher, T. G., & Robertson, S. P. (Eds.). (1991). *Empirical Studies of Programmers: Fourth Workshop*. New Brunswick, NJ: Ablex.
Krasner, H., Curtis, B., & Iscoe, N. (1987). Communication breakdowns and boundary spanning activities on large programming projects. In G. Olson, S. Sheppard & E. Soloway (Eds.), *Empirical Studies of programmers: Second Workshop* (pp. 47-64). Norwood, NJ (USA): Ablex.
Krueger, C. W. (1989). *Models of reuse in software engineering* (Report No. CMU-CS-89-188): Carnegie Mellon University.
Kulkarni, D., & Simon, H. A. (1988). The processes of scientific discovery: The strategy of experimentation. *Cognitive Science, 12*, 139-175.
Kyng, M., & Greenbaum, J. (1991). *Design at work*. Hillsdale, NJ: Erlbaum.
Lange, B. M., & Moher, T. (1989). *Some strategies for reuse in an object-oriented programming environment.* Paper presented at the Proceeding of CHI'89, Austin, USA.
Langley, P., Simon, H. A., Bradshaw, G. L., & Zytkow, J. M. (1987). *Scientific discovery. Computational explorations of the creative processes*. Cambridge, MA: The MIT Press.
Lave, J. (1988). *Cognition in practice: Mind, mathematics and culture in everyday life*. Cambridge, UK: Cambridge University Press.
Lave, J., & Wenger, E. (1991). *Situated learning: legitimate peripheral participation*. Cambridge, UK: Cambridge University Press.
Lawson, B. R. (1979/1984). Cognitive strategies in architectural design. In N. Cross (Ed.), *Developments in design methodology* (pp. 209-220). Chichester: Wiley (Originally published in *Ergonomics*, 1979, *22* (1), 59-68.).
Le Ny, J.-F. (1989a). Questions ouvertes sur la localisation. *Intellectica, 2*(8), 61-84.
Le Ny, J.-F. (1989b). *Science cognitive et comprehension du langage*. Paris: Presses Universitaires de France.

Le Ny, J. F. (1979). *La sémantique psychologique*: Presses Universitaires de France.
Lebahar, J. C. (1983). *Le dessin d'architecte. Simulation graphique et réduction d'incertitude*. Roquevaire (France): Editions Parenthèses.
Leplat, J. (1981). Task analysis and activity analysis in situations of field diagnosis. In J. Rasmussen & W. B. Rouse (Eds.), *Human Detection and Diagnostic of System Failures* (pp. 289-300). New York: Plenum Press.
Leplat, J. (Ed.). (1992a). *L'analyse du travail en psychologie ergonomique. Recueil de textes* (Vol. 2). Toulouse (France): Octarès.
Leplat, J. (Ed.). (1992b). *L'analyse du travail en psychologie ergonomique. Recueil de textes* (Vol. 1). Toulouse: Octarès.
Letovsky, S. (1986). *Cognitive Processes in Program Comprehension.* Paper presented at the Empirical Studies of Programmers: 1st Workshop, Norwood, NJ (USA).
Lieber, J., & Napoli, A. (1997). *Planification à partir de cas et classification.* Paper presented at the JICAA'97, Journées ingénierie des connaissances et apprentissage automatique, Roscoff.
Lindemann, U. (Ed.). (2003). *Human behaviour in design*. Berlin: Springer.
Logan, B., & Smithers, T. (1993). Creativity and design as exploration. In Gero & M. L. Maher (Eds.), *Modeling creativity nad knowledge-based design* (pp. 193-175). Hillsdale, NJ: Erlbaum.
Löwgren, J. (1995). Applying design methodology to software development. In *DIS 95* (pp. 87-95).
Malhotra, A., Thomas, J. C., Carroll, J. M., & Miller, L. A. (1980). Cognitive processes in design. *International Journal of Man-Machine Studies, 12*, 119-140.
Marples, D. L. (1961). The decisions of engineering design. *IRE Transactions on Engineering Management, June*, 55-70.
Martin, G., Détienne, F., & Lavigne, E. (2001, July 9-13). *Analysing viewpoints in design through the argumentation process.* Paper presented at the Interact 2001, Tokyo, Japan.
Mayer, R. E. (1989). Human nonadversary problem solving. In K. J. Gilhooly (Ed.), *Human and machine problem solving* (pp. 39-56). New York: Plenum.
McMahon, T. (1988). *Is Reflective Practice Synonymous with Action Research?*
McNeill, T., Gero, J. S., & Warren, J. (1998). Understanding conceptual electronic design using protocol analysis. *Research in Engineering Design, 10*, 129–140.
Michard, A. (1982). Graphical Presentation of Boolean Expressions in a Database Query Language: Design Notes and an Ergonomic Evaluation. *Behaviour and Information Technology, 1*(3), 279-288.
Millen, D. R. (2000). Rapid ethnography: Time deepening strategies for HCI field research. In *DIS'00* (pp. 280-286).
Miller, G. A., Galanter, E., & Pribram, K. H. (1960). *Plans and the structure of behaviour*. New York: Holt.
Minsky, M. (1961). Steps toward artificial intelligence. *Proceedings IRE, 49*, 8-30.
Morais, A., & Visser, W. (1985). *Etude exploratoire de la programmation d'automates industriels chez des élèves de l'enseignement technique* (Rapport de recherche No. 404). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.
Nakakoji, K., Yamamoto, Y., Takada, S., & Reeves, B. N. (2000). Two-dimensional spatial positioning as a means for reflection in design. In *DIS '00* (pp. 145-154).
Nardi, B. A. (1996a). Studying context: a comparison of activity theory, situated action models, and distributed cognition. In B. A. Nardi (Ed.), *Context and consciousness. Activity theory and human-computer interaction (2nd printing 1997)* (pp. 69-102). Cambridge, MA: The MIT Press.

Nardi, B. A. (Ed.). (1996b). *Context and consciousness. Activity theory and human-computer interaction (2nd printing 199)*. Cambridge, MA: The MIT Press.
Navinchandra, D. (1991). *Exploration and innovation in design*. New-York: Springer.
Newell, A. (1969). Heuristic programming: Ill structured problems. In J. Aronovsky (Ed.), *Progress in operations research*. New York: Wiley.
Newell, A. (1990). *nified theories of cognition*. Cambridge, MA: Harvard University Press.
Newell, A., Shaw, J. C., & Simon, H. A. (1957a). *Empirical explorations of the logic theory machine A case study in heuristics*. Paper presented at the Western Joint Computer Conference.
Newell, A., Shaw, J. C., & Simon, H. A. (1957b). Problem solving in humans and computers. *Carnegie echnical, 21*(4), 35-38.
Newell, A., Shaw, J. C., & Simon, H. A. (1958). Elements of a theory of human problem solving. *Psychological eview, 6*, 151-166.
Newell, A., & Simon, H. A. (1956). The logic theory machine. *IE ransactions on Information heory, I-2*(3), 61-79.
Newell, A., & Simon, H. A. (1972). *Human problem solving*: Prentice Hall.
Newell, A., & Simon, H. A. (1976). Computer science as empirical inquiry: Symbols and search. [1975 ACM Turing Award lecture.]. *Communications of the Association for Computing achinery, 19*(3), 113-126.
Newman, M. W., & Landay, J. A. (2000). Sitemaps, storyboards, and specifications: A sketch of web site design practice. In *DIS'* (pp. 263-274).
Nii, H. P. (1986a). Blackboard systems: the blackboard model of problem solving and the evolution of blackboard architectures. Part One. *he AI aga ine, Summer*.
Nii, H. P. (1986b). Blackboard systems: the blackboard model of problem solving and the evolution of blackboard architectures. Part Two. *he AI aga ine, August*.
Nisbett, R. E., & Wilson, T. D. (1977). Telling more than we can know: Verbal reports on mental processes. *Psychological eview, 8*, 231-259.
Norman, D. (1996). Design as practiced. In Winograd (Ed.), *Bringing design to software* (pp. 233-247). New York: ACM Press.
Norman, D. A. (1993). Cognition in the head and in the world: An introduction to the special issue on situated action. *Cognitive Science, 1*(1), 1-6.
Norman, D. A., & Draper, S. W. (Eds.). (1986). *ser centered system design. ew perspectives on human-computer interaction*. New Jersey: Erlbaum.
Okada, T., & Simon, H. A. (1997). Collaborative discovery in a scientific domain. *Cognitive Science, 21*(2), 109-146.
Olson, S., Sheppard, & Soloway, E. (Eds.). (1987). *Empirical Studies of Programmers Second orkshop*. Norwood, NJ (USA): Ablex.
Pahl, G., Badke-Schaub, P., & Frankenberger, E. (1999). Resume of 12 years interdisciplinary empirical studies of engineering design in Germany. *Design Studies, 2*(5), 481-494.
Pahl, G., & Beitz, W. (1977). *onstruktionslehre*. Berlin: Springer Verlag.
Pahl, G., & Beitz, W. (1984). *Engineering design* (K. M. Wallace, Trans.). London: The Design Council.
Pahl, G., & Beitz, W. (1996). *Engineering design. A systematic approach* (K. M. Wallace, Blessing & F. Bauert, Trans. 2nd, enlarged and updated ed.). London: Springer.
Pahl, G., Frankenberger, E., & Badke-Schaub, P. (1999). Historical background and aims of interdisciplinary research between Bamberg, Darmstadt and Munich. *Design Studies, 2*(5), 401-406.
Pakman, M. (2000). Thematic Foreword: Reflective Practices: The Legacy Of Donald Schön. *Cybernetics and Human nowing,* (2-3), 5-8.
Perrin, J. (1999). Diversité des représentations du processus de conception, diversité des modes de pilotage de ces processus. In J. Perrin (Ed.), (pp. 19-39).

Perrin, J. (Ed.). (2002). *International Conference In Honour of Herbert Simon "he Sciences of Design. he Scientific Challenge for the 21st Century", yon (rance), ISA, 1-16 arch 22.*

Pitrat, J. (2002). Herbert Simon, pilonnier de l'Intelligence Artificielle. *evue d'Intelligence Artificielle, 16*(1-2), 11-16.

Pitrat, J. (Ed.). (2002). *evue d'Intelligence Artificielle, 16 (1-2), epr sentations, d couverte et rationalit Hommage Herbert Simon.*

Pu, P. (1993). Introduction: Issues in case-based design systems. *AI EDA , *(2), 79-85.

Qin, Y., & Simon, H. A. (1990). Laboratory replication of scientific discovery processes. *Cognitive Science, 1*, 281-312.

Rasmussen, J. (1979). *On the structure of knowledge. A morphology of mental models in a man-machine system context* (Report Ris-M-2192). Roskilde (Danmark): Ris National Laboratory.

Reimann, P., & Chi, M. T. H. (1989). Human expertise. In K. J. Gilhooly (Ed.), *Human and machine problem solving* (pp. 161-191). New York: Plenum.

Reitman, W. (1964). Heuristic decision procedures, open constraints, and the structure of ill-defined problems. In M. W. Shelley & G. L. Bryan (Eds.), *Human udgments and optimality*. New York: Wiley.

Reitman, W. (1965). *Cognition and thought*. New York: Wiley.

Richard, J. F. (1990). *es activit s mentales. Comprendre, raisonner, trouver des solutions.*: Armand Colin.

Rist, R. S. (1990). Variability in program design: the interaction of process with knowledge. *International ournal of an- achines Studies, 33 (3)*(special issue "what programmers know"), 305-322.

Rist, R. S. (1991a). Knowledge creation and retrieval in program design: a comparison of novice and intermediate students programmers. *Human-Computer Interactions, 6*, 1-46.

Rist, R. S. (1991b, 25 August). *odels of routine and non-routine design in the domaine of programming.* Paper presented at the A.I. in Design. Workshop of the 12th international conference on A.I., Sydney (Australia).

Rittel, H. W. J. (Interviewed by Grant and Protzen, 1972/1984). Second-generation design methods. In N. Cross (Ed.), *Developments in design methodology* (pp. 317-327). Chichester: Wiley (Originally published in *he D th Anniversary eport D Occasional Paper 1*, 1972, pp. 5-10).

Rittel, H. W. J., & Webber, M. M. (1973/1984). Planning problems are wicked problems. In N. Cross (Ed.), *Developments in design methodology*. Chichester: Wiley (Originally published as part of "Dilemmas in a general theory of planning", *Policy Sciences*, 1973, , 155-169).

Robert, J. M. (1979). *es r sultats des recherches sur le processus de conception, les comportements et les processus cognitifs mis en eu* (Rapport INRIA No. EC 7912 R01). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Rogers, Y., & Scaife, M. External Cognition.

Rosenfield, I. (1988). *he invention of memory*. New York, NY: Basic Books.

Rosson, M. B., & Alpert, S. R. (1990). The Cognitive Consequences of object-oriented Design. *Human-Computer Interaction, *, 345-379.

Rowe, P. G. (1987). *Design thinking*. Cambridge, MA: The MIT Press.

Scacchi, W. (2001). Process models in Software Engineering. In J. J. Marciniak (Ed.), *Encyclopedia of Software Engineering* (2 ed.). New York: Wiley.

Schön, D. A. (1983). *he reflective practitioner How professionals think in action*. New York: Basic Books (reprinted in 1995).

Schön, D. A. (1984). *he design studio An exploration of its traditions and potentials*. London: RIBA.

Schön, D. A. (1987a). Educating the reflective practicioner. Washington, DC.: American Educational Research Association.
Schön, D. A. (1987b). *Educating the reflective practicioner*. San Francisco: Jossey-Bass.
Schön, D. A. (1988). Designing: Rules, types and worlds. *Design Studies, 9*(3), 181-190.
Schön, D. A. (1992, March). Designing as reflective conversation with the materials of a design situation. *nowledge-Based Systems,* (1), 3-14.
Schön, D. A., & Wiggins, G. (1992, Spring). Kinds of seeing and their functions in designing. *Design Studies, 13*(2), 135-156.
Sharpe, J. E. E. (1995). Computer tools for integrated conceptual design. *Design Studies, 16*(4), 471-488.
Simon, H. A. (1955). A behavioral model of rational choice. *uarterly ournal of Economics, 69*, 99-118.
Simon, H. A. (1971/1975). Style in design. In Eastman (Ed.), *Spatial synthesis in computer-aided building design* (pp. 287-309). London: Applied Science Publishers.
First published in J. Archea & Eastman (Eds.) (1971). *EDA O, Proceedings 2nd Ann. Environmental Design esearch Association Conference, 1-1, October 19* (pp. 1-10). Stroudsbury, PA: Dowden, Hutchinson & Ross, Inc. (Version of Simon's text referred to in this paper).
Simon, H. A. (1973/1984). The structure of ill-structured problems. *Artificial Intelligence, 193, *, 181-201.
Also in N. Cross (Ed.) (1984), *Developments in design methodology* (pp. 1145-1166). Chichester: Wiley.
Simon, H. A. (1977a). *odels of discovery*. Boston: Reidel.
Simon, H. A. (1977b). The next hundred years: Engineering design. In L. E. Jones (Ed.), *he next hundred years* (pp. 89-104). Toronto: University of Toronto, Faculty of Applied Science and Engineering.
Simon, H. A. (1978). Information-processing theory of human problem solving. In W. K. Estes (Ed.), *Handbook of learning and cognitive processes* (Vol. V. Human information processing, pp. 271-295). Hillsdale, NJ: Erlbaum.
Simon, H. A. (1979). Information processing models of cognition. *Annual eview of Psychology, 3*, 363-396.
Simon, H. A. (1980). Technology: Source of opportunity and constraint in design. *College of Design, Architecture, and Art ournal, 1*, 26-33.
Simon, H. A. (1982). *odels of bounded rationality* (Vol. 1 & 2). Cambridge, MA: The MIT Press.
Simon, H. A. (1984/2002). Sur le colloque Sciences de l'Intelligence, Sciences de l'Artificiel. Extraits des commentaires et des réponses aux questions. *evue d'Intelligence Artificielle, 16*(1-2), 39-52. [Le colloque *Sciences de l Intelligence, Sciences de l Artificiel* s'est tenu en 1984].
Simon, H. A. (1987/1995). Problem forming, problem finding, and problem solving in design. In A. Collen & W. W. Gasparski (Eds.), *Design and systems eneral applications of methodology* (Vol. 3, pp. 245-257). New Brunswick, NJ: Transaction Publishers. (Text of a lecture delivered to the First International Congress on Planning and Design Theory, Boston, 1987).
Simon, H. A. (1992a). Alternative representations for cognition: Search and reasoning. In J. H.L. Pick, P. v. d. Broek & D. C. Knill (Eds.), *Cognition Conceptual and methodological issues* (pp. 121-142). Washington, DC: American Psychological Association.
Simon, H. A. (1992b). Scientific discovery as problem solving. *International Studies in the Philosophy of Science, 6*, 3-14.
Simon, H. A. (1992c). Scientific discovery as problem solving: reply to critics. *International Studies in the Philosophy of Science, 6*, 69-88.

Simon, H. A. (1995). *Explaining the ineffable□AI on the topics of intuition, insight and inspiration.* Paper presented at the Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence.

Simon, H. A. (1997). Integrated design and process technology. □*ournal of Integrated Design and Process Science, 1*(1), 9-16.

Simon, H. A. (1997). □*odels of bounded rationality* (Vol. 3). Cambridge, MA: The MIT Press.

Simon, H. A. (1999). □*he sciences of the artificial (3rd, rev. ed. 1996□Orig. ed. 1969□2nd, rev. ed. 1981)* (3 ed.). Cambridge, MA: The MIT Press.

Simon, H. A. (2000). Bounded rationality in social science: Today and tomorrow. □*ind □ Society, 1*(1), 25-39.

Simon, H. A. (2001a). Creativity in the arts and the sciences. In Cultures of creativity: The centennial celebrations of the Nobel Prizes. □*he □enyon □eview, Spring, 23*(2), 203-220.

Simon, H. A. (2001b). "Seek and ye shall find" How curiosity engenders discovery. In K. D. Crowley, C. D. Schunn & T. Okada (Eds.), *Designing for science□Implications from everyday classroom, and professional settings* (pp. 3-18). Mahwah, NJ: Erlbaum.

Simon, H. A., & Newell, A. (1956). Models: Their uses and limitations. In L. D. White (Ed.), □*he state of the social sciences* (pp. 66-83). Chicago: University of Chicago Press.

Smith, M. K. (July 2001, July 02, 2001). *donald schon (sch□n)□learning, reflection and change*. Retrieved August 23, 2001, 2001, from Retrieved August 23, 2001, from http://www.infed.org/thinkers/et-schon.htm

Smyth, B., & Keane, M. T. (1995). Some experiments on adaptation-guided retrieval. In M. Veloso & A. Aamodt (Eds.), *Case-based reasoning, □esearch and development, □irst International Conference, Proceedings of ICCB□-9□*(pp. 313-324): Springer.

Soloway, E., & Iyengar, S. (Eds.). (1986). *Empirical Studies of Programmers□□irst workshop*. Norwood, NJ (USA): Ablex.

Special issue on Situated action. (1993). *Cognitive Science, 1□*(1).

Sperandio, J.-C. (1980). □*a psychologie en ergonomie*. Paris: PUF.

Stefik, M. J. (1981a). Planning and meta-planning (MOLGEN: Part 2). *Artificial Intelligence, 16*, 141-170.

Stefik, M. J. (1981b). Planning with constraints (MOLGEN: Part 1). *Artificial Intelligence, 16*, 111-140.

Steinberg, L. I. (1987). *Design as refinement plus constraint propagation□the □E□ED experience.* Paper presented at the Sixth National Conference on Artificial Intelligence (AAAI-87), Los Altos, CA.

Stolterman, E. (1992). *How system designers think. About design and methods. Some □eflections Based on an Interview Study*. Retrieved October, 16, 2002, from http://iris.informatik.gu.se/sjis/vol4/stolter.shtml

Suchman, L. A. (1990 (reprint of the 1st Ed., 1987)). *Plans and situated actions. □he problem of human-machine communication*. Cambridge: Cambridge University Press.

Suchman, L. A. (1993). Response to Vera and Simon's situated action: A symbolic interpretation. *Cognitive Science, 1□*(1), 71-77.

Suwa, M., & Tversky, B. (1997). What do architects and students perceive in their design sketches? A protocol analysis. *Design Studies, 18*(4), 385-404.

Thomas, J. C. (1989). Problem solving by human-machine interaction. In K. J. Gilhooly (Ed.), *Human and machine problem solving* (pp. 317-362). New York: Plenum.

Thomas, J. C., & Carroll, J. M. (1979/1984). The psychological study of design. In N. Cross (Ed.), *Developments in design methodology* (pp. 221-235). Chichester: Wiley (Originally published in *Design Studies,* 1979*, 1* (1), 5-11).

Thunem, S., & Sindre, G. (1992, October 29-30 1992). *Development with and for reuse.* Paper presented at the ERCIM 92 "Methods and Tools for Software Reuse", Heraklion.

Traverso, V., & Visser, W. (2003). Confrontation de deux méthodologies d'analyse de situations d'élaboration collective de solution. In J. M. C. Bastien (Ed.), *Actes des Deuxi□-*

mes □ourn□es d'Etude en Psychologie ergonomique - EPI□□E 2□□3 (Boulogne-Billancourt, □rance, 2-3 octobre) (pp. 241-246). Rocquencourt (France): INRIA.

Ullman, D., Dietterich, T. G., & Staufer, L. A. (1988). A model of the mechanical design process based on empirical data. *AI EDA□, 2*, 33-52.

Ullman, D., Staufer, L. A., & Dietterich, T. G. (1987). Toward expert CAD. *Computers in □e-chanical Engineering, 6*(3), 56-70.

Ullman, D. G., & Culley, S. J. (1994). The mechanical design process. *Design Studies, 1□*(1), 115.

VDI. (August 1987). *Systematic Approach to the Design of □echnical Systems and Products, □uideline □DI 2221* (K. M. Wallace, Trans.). Berlin: Beuth Verlag.

Vera, A. H., & Simon, H. A. (1993a). Situated action: A symbolic interpretation. *Cognitive Science, 1□*(1), 7-48.

Vera, A. H., & Simon, H. A. (1993b). Situated action: Reply to reviewers. *Cognitive Science, 1□*(1), 77-86.

Vera, A. H., & Simon, H. A. (1993c). Situated action: Reply to William Clancey. *Cognitive Science, 1□*, 117-133.

Vermersch, P. (1994). *□'entretien d'explicitation*. Paris: ESF.

Visser, W. (1985). *□od□lisation de l'activit□ de programmation de syst□mes de commande □□odeling the activity of programming control systems□(In □rench).* Paper presented at the Actes du colloque COGNITIVA 85 (Tome 2), Paris.

Visser, W. (1986a). *Activit□ de conception d'une installation automatis□e. 1. □a construction du sch□ma de fonctionnement. 1a. □e sch□ma de s□quences.* (Rapport PPE). Rocquencourt (France): Institut National de Recherche en Informatique et Automatique.

Visser, W. (1986b). *Etude de deux □tapes de la conception □□criture et mise au point d'un programme* (Rapport PPE). Rocquencourt (France): Institut National de Recherche en Informatique et Automatique.

Visser, W. (1986c). *Etude pr□liminaire du travail du programmeur d'automate programmable dans un bureau d'□tudes □lectrique* (Rapport PPE). Rocquencourt (France): Institut National de Recherche en Informatique et Automatique.

Visser, W. (1987). Strategies in programming programmable controllers: a field study on a professional programmer. In G. Olson, S. Sheppard & E. Soloway (Eds.), *Empirical Studies of programmers□Second □orkshop* (pp. 217-230). Norwood, NJ (USA): Ablex.

Visser, W. (1988a). *□iving up a hierarchical plan in a design activity* (Rapport de recherche No. n° 814). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique. English version of W. Visser (1987, 18-22 mai). *Abandon d'un plan hierarchique dans une activite de conception.* Actes du Colloque scientifique MARI 87 Machines et Réseaux Intelligents - COGNITIVA 87 (Tome 1), Paris, La Villette.

Visser, W. (1988b). L'activité de comparaison de représentations dans la mise au point de programmes. *□e □ravail Humain, □um□ro Sp□cial "Psychologie ergonomique de la programmation informatique", □1*(4), 351-362.

Visser, W. (1988c, 14-16 June). *□owards modelling the activity of design□an observational study on a specification stage.* Paper presented at the Proceedings of the IFAC/IFIP/IEA/IFORS Conference on Man-Machine Systems. Analysis, Design and Evaluation, Oulu (Finland).

Visser, W. (1989). *□he opportunistic use of a plan in a design activity□an empirical study of specification* (Rapport de recherche No. N° 1035). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Visser, W. (1990). More or less following a plan during design: opportunistic deviations in specification. *International □ournal of □an-□achine Studies. Special issue□□hat programmers know, 33*, 247-278.

Visser, W. (1991a). The cognitive psychology viewpoint on design: examples from empirical studies. In Gero (Ed.), *Artificial Intelligence in Design '91*. Oxford: Butterworth-Heinemann.

Visser, W. (1991b). Evocation and elaboration of solutions: Different types of problem-solving actions. An empirical study on the design of an aerospace artifact. In T. Kohonen & F. Fogelman-Soulié (Eds.), *□hird CO□□I□I□A symposium. CO□□I□I□A 9□ At the crossroads of Artificial Intelligence, Cognitive science, and □euroscience, Paris* (pp. 689-696). Amsterdam: Elsevier.

Visser, W. (1991c). Planning in routine design. Some counterintuitive data from empirical studies. In Gero & F. Sudweeks (Eds.), *Preprints of the "Artificial Intelligence in Design" □orkshop of the □welfth International □oint Conference on Artificial Intelligence, Sydney (Australia), 2□August*. Sydney (Australia): University of Sydney.

Visser, W. (1992). Designers' activities examined at three levels: organization, strategies & problem-solving. *□nowledge-Based Systems, □*(1), 92-104.

Visser, W. (1993a). Collective design: A cognitive analysis of cooperation in practice. In N. F. M. Roozenburg (Ed.), *Proceedings of ICED 93, 9th International Conference on Engineering Design* (Vol. 1, pp. 385-392). □ürich: HEURISTA.

Visser, W. (1993b). Design & knowledge modeling - Knowledge modeling as design. *Communication and Cognition - Artificial Intelligence, 1□*(3), 219-233.

Visser, W. (1994a). Organisation of design activities: opportunistic, with hierarchical episodes. *Interacting □ith Computers, 6*(3), 239-274 (Executive summary: 235-238).

Visser, W. (1994b). Planning and organization in expert design activities. In Gilmore, R. Winder & Détienne (Eds.), *□ser-centred requirements for software engineering environments*. Berlin: Springer.

Visser, W. (1994c, September 20-22). *□se of episodic knowledge in design problem solving.* Paper presented at the Paper presented at the Delft Protocols Workshop "Analysing Design Activity", Delft (The Netherlands).

Visser, W. (1994d). Workshop report of the IJCAI'93: Thirteenth International Joint Conference on Artificial Intelligence workshop on 'Reuse of designs: an interdisciplinary cognitive approach'. *AI Communications, □*(1), 64-65.

Visser, W. (1995a). Reuse of knowledge: empirical studies. In M. Veloso & A. Aamodt (Eds.), *Case-based reasoning research and development* (pp. 335-346). Berlin: Springer.

Visser, W. (1995b). Use of episodic knowledge and information in design problem solving. *Design Studies, 16*(2), 171-187.
also in N. Cross, H. Christiaans & K. Dorst (Eds.) (1996), *Analysing design activity* (Ch. 1913, pp. 1271-1289). Chichester: Wiley.

Visser, W. (1996). Two functions of analogical reasoning in design: a cognitive-psychology approach. *Design Studies, 1□*, 417-434.

Visser, W. (1999a, 20-22 Octobre). *Conception individuelle et collective. Approche de l'ergonomie cognitive.* Paper presented at the Université d'Autome PRIMECA (Pôle de Ressources Informatiques pour la Mécanique) sur le thème "Modélisation des processus de conception", session "Capitalisation et utilisation des connaissances", Nancy (France).

Visser, W. (1999b). Etudes en ergonomie cognitive sur la réutilisation en conception: quelles le□ons pour le raisonnement à partir de cas? *□evue d'Intelligence Artificielle, 13*, 129-154.

Visser, W. (2002a). Plans : Etude de la planification de parcours en ville (Projet n° COG 54 du Programme Cognitique, thème "Cognition spatiale"). Rapport de fin de recherche. In *Cognitique. Action concert□e incitative 1999-2□□2. □□sum□s des pro□ets de recheche soutenus*. In (pp. 87-89). Paris (France): Cognitique.

Visser, W. (2002b, 15-16 March 2002). *A □ribute to Simon, and some □too late□ questions, by a cognitive ergonomist.* Paper presented at the International Conference In Honour

of Herbert Simon "The Sciences of Design. The Scientific Challenge for the 21st Century", Lyon (France): INSA.
A slightly modified version of this text has been published as Research report N° 4462. Rocquencourt: INRIA. May 2002.

Visser, W. (Ed.). (1993c). *Proceedings of the I□CAI'93 □hirteenth International □oint Conference on Artificial Intelligence □orkshop "□euse of designs□an interdisciplinary cognitive approach", Chamb□ry (□rance), August 29, 1993*. Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Visser, W., & Bonnardel, N. (1989). *□a r□solution de probl□mes lors de la conception d'une antenne. Analyse de l'activit□* (Rapport PPE). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Visser, W., & Falzon, P. (1988). *Eliciting expert knowledge in a design activity□some methodological issues* (Rapport de recherche No. N° 906). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Visser, W., & Falzon, P. (1989, 18-22 September). *Eliciting the knowledge of a design expert.* Paper presented at the Abridged proceedings. Poster sessions of HCI International '89 - Third International Conference on Human-Computer Interaction, Boston, MA (U.S.A.).

Visser, W., & Hoc, J.-M. (1990). Expert software design strategies. In Hoc, T. Green, R. Samur□ay & Gilmore (Eds.), *Psychology of programming* (pp. 235-250). London: Academic Press.

Visser, W., & Morais, A. (1988). *Concurrent use of different techniques for gathering data on the programming activity* (Rapport de recherche No. n° 939). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Visser, W., & Morais, A. (1991). Concurrent use of different expertise elicitation methods applied to the study of the programming activity. In M. J. Tauber & D. Ackermann (Eds.), *□ental models and human-computer interaction* (Vol. 2). Amsterdam: Elsevier.

Visser, W., & Perron, L. (1996a, 10-13 September). *□oal-oriented categorisation□the role of task, daily activity, expertise and problem attributes.* Paper presented at the Proceedings of ECCE'96 (Eight European Conference on Cognitive Ergonomics), Granada (Spain).

Visser, W., & Perron, L. (1996b). *□he organisation of professional operative knowledge□goal-oriented categorisations.* Paper presented at the Proceedings of PPIG-8 (8th Annual Workshop of the Psychology of Programming Interest Group) (Abstract), Ghent (Belgium): KaHo Sint Lieven.

Visser, W., & Richard, J.-F. (1984). *Etude pr□liminaire de l'activit□ de programmation d'automates programmables* (Rapport PPE). Rocquencourt (France): Institut National de Recherche en Informatique et Automatique.

Visser, W., & Trousse, B. (1993). Reuse of designs: desperately seeking an interdisciplinary cognitive approach. In W. Visser (Ed.), *Proceedings of the I□CAI □hirteenth International □oint Conference on Artificial Intelligence □orkshop "□euse of designs□an interdisciplinary cognitive approach", Chamb□ry (□rance), August 29, 1993* (pp. 1-14 of 11-13). Rocquencourt (France): Institut National de Recherche en Informatique et en Automatique.

Voss, J. F., Greene, T. R., Post, T. A., & Penner, B. C. (1983). Problem-solving skill in the social sciences. In G. Bower (Ed.), *□he psychology of learning and motivation* (Vol. 17). New York: Academic Press.

Walz, D. B., Elam, J. J., Krasner, H., & Curtis, B. (1987). A methodology for studying software design teams: an investigation of conflict behaviors in the requirements definition phase. In G. Olson, S. Sheppard & E. Soloway (Eds.), *Empirical Studies of programmers□Second □orkshop* (pp. 83-99). Norwood, NJ (USA): Ablex.

Warren, C., & Whitefield, A. (1987). The role of task characterization in transferring models of users: the example of engineering design. In H.-J. Bullinger & B. Shackel (Eds.), *Human-computer interaction - I□□E□AC□'8□* Amsterdam: North-Holland.

Weber, G. (1991). *Explanation-based retrieval in a case-based learning model.* Paper presented at the Thirteenth Annual Meeting of the Cognitive Science Society.

Whitefield, A. (1986). An analysis and comparison of knowledge use in designing with and without CAD. In A. Smith (Ed.), *□nowledge engineering and computer modelling in CAD. Proceedings of CAD86. Seventh International Conference on the Computer as a Design □ool*. London: Butterworths.

Whitefield, A. (1989). Constructing appropriate models of computer users: the case of engineering designers. In J. Long & A. Whitefield (Eds.), *Cognitive ergonomics and human-computer interaction*. Cambridge, MA: Cambridge University Press.

Wiedenbeck, S., & Scholtz, J. (Eds.). (1997). *Empirical Studies of Programmers□Seventh □orkshop*. New York (NY) USA: ACM Press.

Winograd, T. (1997). *□rom computing machinery to interaction design*.

Winograd, T. (Ed.). (1996). *Bringing design to software*. New York: ACM Press.

Winograd, T., & Flores, F. (1986). *□nderstanding computers and cognition□A new foundation for design*. Norwood, NJ: Ablex.

Wirth, N. (1971). Program development by stepwise refinement. *Communications of the AC□, 1□*(4), 221-227.

Yourden, E., & Constantine, L. (1978). *Structured design*: Yourden Press.

*□*hang, J., & Norman, D. A. (1994). Representations in distributed cognitive tasks. *Cognitive Science, 18*, 87-122.